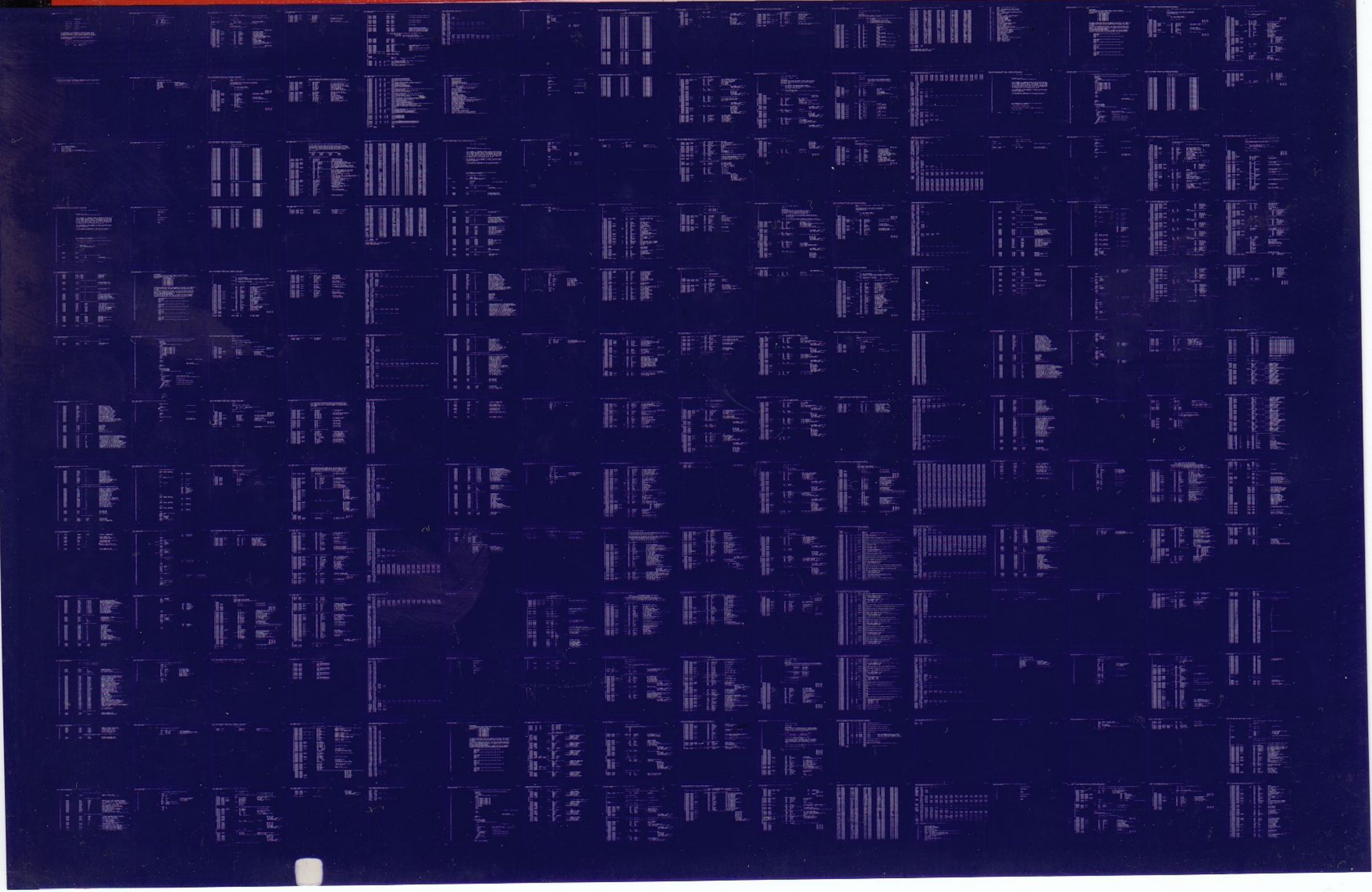
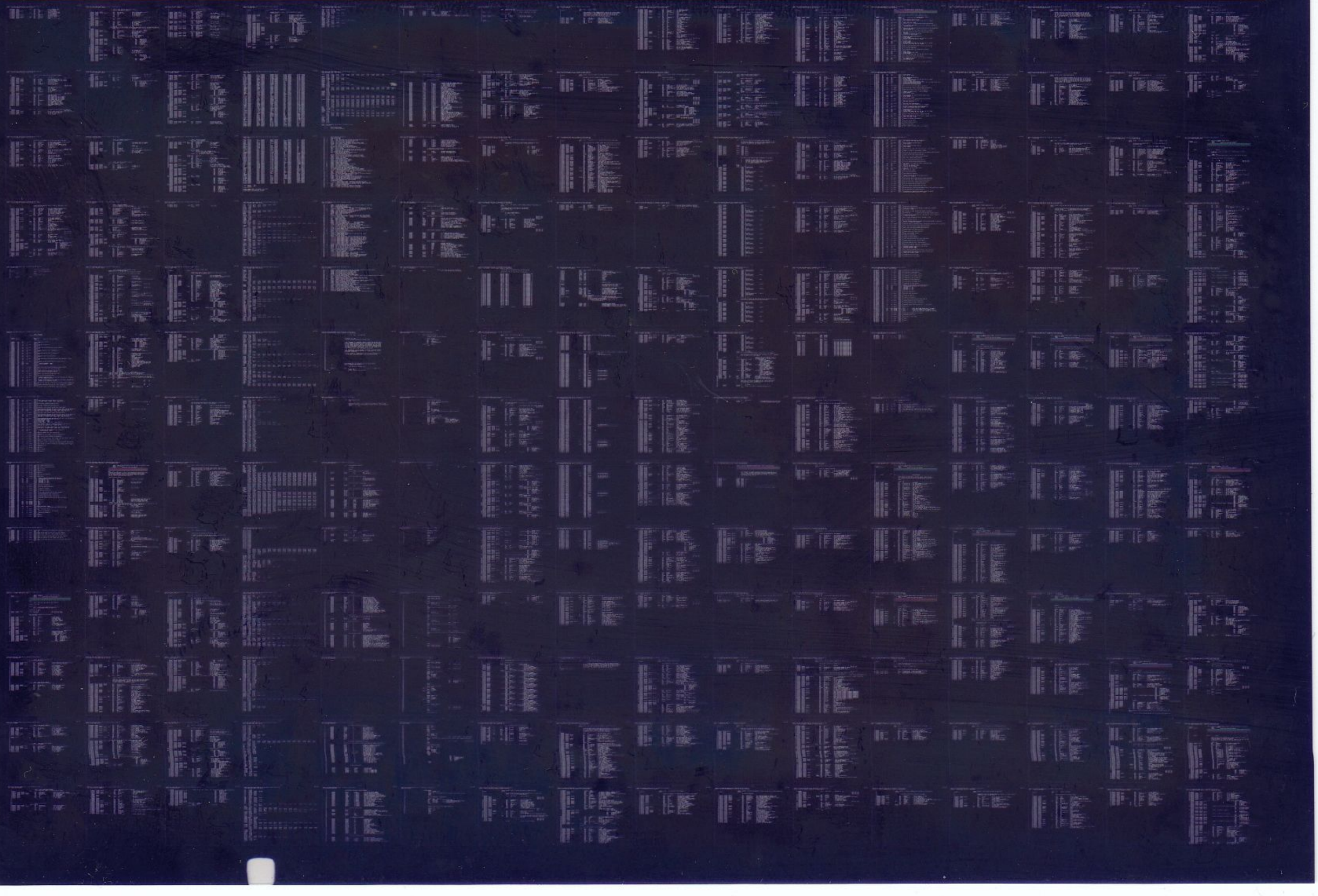
RA80

UDA DIAG DATA FILE
CZUDDB0

AH-S834B-MC
FICHE 1 OF 3



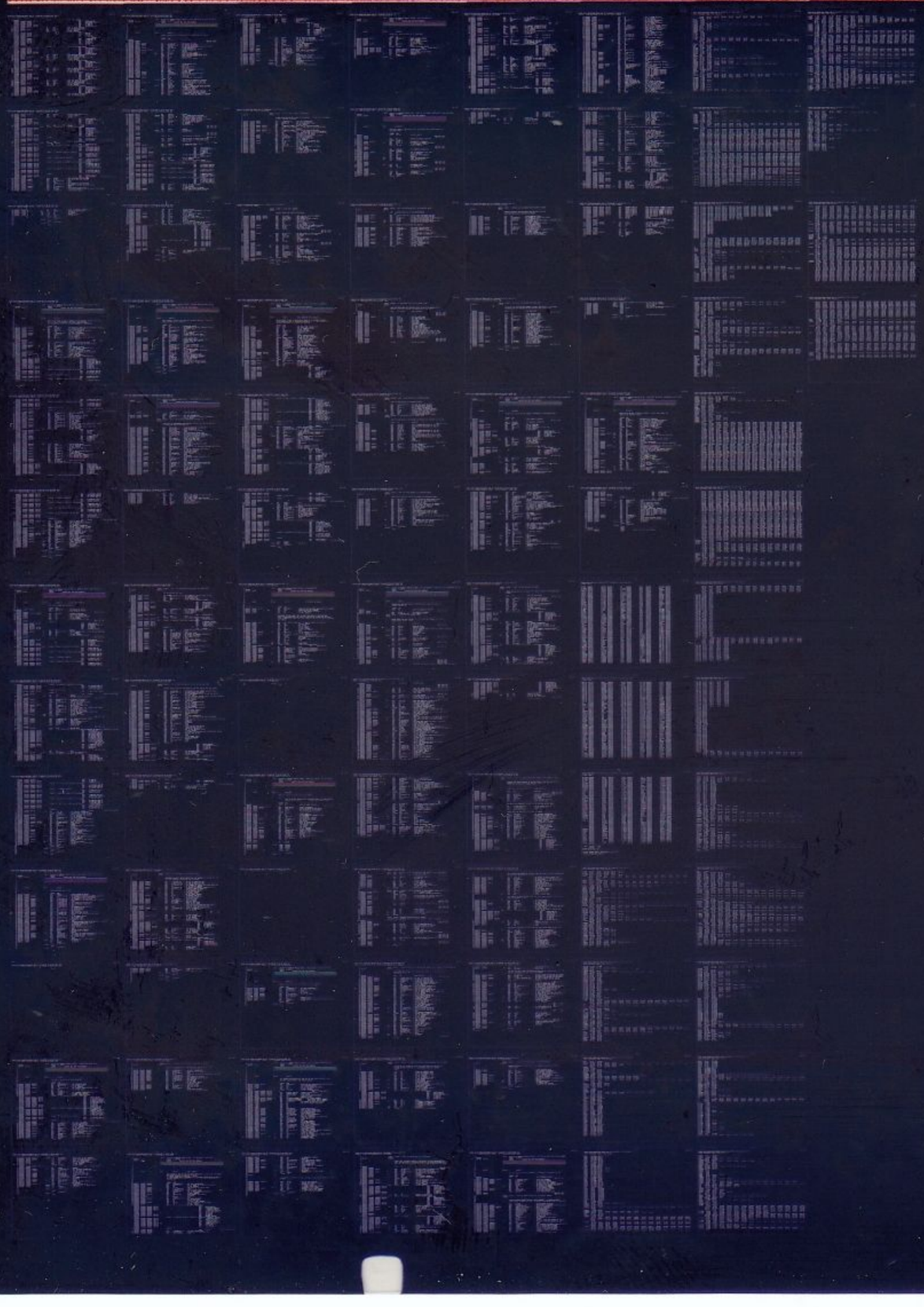
RA80 UDA DIAG DATA FILE AH-S834B-MC SEP 1982
CZUDDB0 FICHE 2 OF 3
MADE IN USA



RA80

UDA DIAG DATA FILE
CZUDDB0

AH-S834B-MC
FICHE 3 OF 3



IDENIFICATION

PRODUCT CODE: AC-S833B-MC

PRODUCT NAME: CZUDDB0 UDA50 DIAG DATA FILE

PRODUCT DATE: 14-APR-82

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: DALE KECK

CZUDDB0 UDA50 DIAG DATA FILE 14-APR-82

These are the listings of the UDA50 drive diagnostics. They are the diagnostics that are downline loaded by the diagnostic program CZUDCB0. Included is the listing for test 1, test 2, test 3, and test 4.

For further information see the documentation that is associated with CZUDCB0.

UDAT1 UNIBUS ADDRESSING DMACR X04.01 13-APR-82 15:48:42
TABLE OF CONTENTS

```
                                .TITLE  UDAT1   UNIBUS ADDRESSING


                        ;
                        ;
                        ; COPYRIGHT (C)  1981
                        ; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.
                        ;
                        ;
                        ; THIS  SOFTWARE  IS  FURNISHED  UNDER A LICENSE FOR USE ONLY ON A
                        ; SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION
                        ; OF  THE  ABOVE COPYRIGHT  NOTICE.  THIS SOFTWARE, OR  ANY  OTHER
                        ; COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE  MADE AVAILABLE
                        ; TO  ANY  OTHER PERSON  EXCEPT  FOR USE ON SUCH SYSTEM AND TO ONE
                        ; WHO AGREES TO THESE LICENSE TERMS.  TITLE  TO  AND  OWNERSHIP OF
                        ; THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DIGITAL.
                        ;
                        ; THE  INFORMATION  IN THIS DOCUMENT IS SUBJECT  TO CHANGE WITHOUT
                        ; NOTICE AND SHOULD NOT BE CONSTRUED AS A  COMMITMENT  BY  DIGITAL
                        ; EQUIPMENT CORPORATION.
                        ;
                        ; DIGITAL ASSUMES NO  RESPONSIBILITY FOR THE USE OR RELIABILITY OF
                        ; ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;THIS PROGRAM SHALL BE ASSEMBLED WITH THE PROGRAM DMACRO
                        ;USING A COMMAND LINE SIMILAR TO:
                        ;
                        ;  UDAT1.BIN,UDAT1/C=[1,2]DMACRO,UDAT1T,UDATP,UDAT1M,UDATR,UDAT1S
                        ;
                        ;       VERSION 1.0 FOR RELEASE
                        ;
        000000          TEST4   =       0                       ; THIS IS NOT TEST4
                        ;       .ENABLE ABS
000000                          DMCODE  UDADM1,0,714,3,0,1000
                        .SBTTL  UDA DM PROGRAM PARAMETERS

                                .LIST MEB
                        ;
                        ;       EQUATES
                        ;
        000001          BIT00 = 1
        000002          BIT01 = 2
                        ;
                        ;       HIGHEST USABLE LOCATION OF UDA MEMORY + 1
                        ;
        010000          HIMEM   =       10000                   ; HIGH MEMORY+1
        007774          OVSTRT  =       7774                    ; OVERLAY ADDRESS LOCATION
                        ;
                        ;       DUP PACKET VALUE
                        ;
        010000          DU.QUE = 10000                          ;QUESTION
```

```
                                DU.DFL = 20000                  ;DEFAULT QUESTION
                                DU.INF = 30000                  ;INFORMATION
                                DU.TER = 40000                  ;TERMINATOR
                                DU.FTL = 50000                  ;FATAL ERROR
                                DU.SPC = 60000                  ;SPECIAL
                                ;
                                ;       OFFSETS FOR FORMAT TRACK TABLE
                                ;
                                FT.BUF  =       0.              ;BUFFER POINTER OFFSET
                                FT.HI   =       1.              ;HI ORDER HEADER OFFSET
                                FT.LOW  =       2.              ;LOW ORDER HEADER OFFSET
                                ;
                                ;       OFFSETS FOR FORMAT TRACK BUFFER
                                ;
                                FB.DAT  =         0.            ;FIRST DATA WORD OFFSET
                                FB.EDC  =       256.            ;EDC WORD OFFSET
                                ;
                                ;       OFFSETS FOR READ/WRITE I/O CHAIN TABLES
                                ;
                                RW.STAT =       0.              ;STATUS AND NEXT BUFFER POINTER OFFSET
                                RW.BUF  =       1.              ;POINTER TO DATA BUFFER
                                RW.LOW  =       2.              ;HI ORDER EXPECTED HEADER
                                RW.HI   =       3.              ;LOW ORDER EXPECTED HEADER
                                RW.CMD  =       4.              ;SDI COMMAND AND HEAD ADDRESS
                                RW.SDI  =       5.              ;DUMMY SDI CONTROL BLOCK POINTER
                                RW.ANG  =       6.              ;THETA FROM INDEX
                                ;
                                ;       CONSTANTS FOR READ AND WRITE XFC'S
                                ;
                                WSTOP   =       140000          ; LAST ENTRY IN CHAIN FOR WRITE
                                WCONT   =       40000           ; WRITE CONTINUE
                                RSTOP   =       100000          ; LAST ENTRY IN CHAIN FOR READ
                                RCONT   =       0               ; READ CONTINUE
                                FSTOP   =       100000          ; LAST ENTRY IN CHAIN FOR FORMAT
                                WREAL   =       122400          ; WRITE REAL TIME ECOMMAND
                                RREAL   =       1?400           ; READ REAL TIME COMMAND
                                ECCFLG  =       10000           ; ECC ERROR IN BUFFER BIT
                                EOC     =       100000          ; END OF CHAIN
                                BUFFLG  =       40000           ; BUFFER FULL OR EMPTY FLAG
                                ;
                                ;       HEADER CODES
                                ;
                                HD.LBN  =       000000          ;GOOD LBN
                                HD.RBN  =       060000          ;GOOD RBN, PERHAPS UNUSED
                                HD.REV  =       030000          ;REVECTORED LBN
                                HD.BAD  =       110000          ;BAD BLOCK
                                HD.PRV  =       050000          ;PRIMARY REVECTORED BLOCK
                                HD.XBN  =       120000          ;XBN BLOCK
                                HD.DBN  =       140000          ;DBN BLOCK

                                ;       OFFSETS FOR DATA BUFFERS

                                BF.DAT  =         0.            ;DATA
                                BF.EDC  =       256.            ;ERROR DETECTION CODE
```

```
    115          000401                  BF.ECC  =       257.                    ;LAST 17 ECC RESIDUES
    116
    117                                  ;       BUFFER AND READ/WRITE CHAIN LINK SIZES
    118
    119          000401                  WBUFLN  =       257.                    ; WRITE BUFFER SIZE
    120          000415                  RBUFLN  =       WBUFLN+12.              ; READ BUFFER SIZE
    121          000007                  LINKLN  =       7.                      ; LINK SIZE
```

```
                                ;       XFC DEFINITION EQUATES
                                ;
000000                          BREAK   =       0.      ;BREAKPOINT XFC CODE
000001                          FORMAT  =       1.      ;FORMAT TRACK XFC CODE
000002                          XREAD   =       2.      ;READ N SECTORS XFC CODE
000003                          XWRITE  =       3.      ;WRITE N SECTORS XFC CODE
000004                          SEND    =       4.      ;SEND SDI COMMAND XFC CODE
000005                          RCV     =       5.      ;RECEIVE SDI MESSAGE XFC CODE
000006                          COMPARE =       6.      ;COMPARE DATA PATTERN TO BUFFER
000007                          STATUS  =       7.      ;RETURN DRIVE STATUS XFC CODE
000010                          ECHO    =       8.      ;ECHO DATA TO DRIVE XFC CODE
000011                          DINIT   =       9.      ;DRIVE INITIALIZE XFC CODE
000012                          WAITSI  =       10.     ;WAIT FOR SECTOR OR INDEX PULSE
000013                          UREAD   =       11.     ;READ UNIBUS MEMORY XFC CODE
000014                          UWRITE  =       12.     ;WRITE UNIBUS MEMORY XFC CODE
000015                          ECC     =       13.     ;DO ECC ON BUFFER XFC CODE
000016                          MRD     =       14.     ;SEND BUFFER TO MAINTENANCE READ COMMAND
000017                          MWR     =       15.     ;GET BUFFER FROM MAINTENANCE WRITE COMMAND
000020                          CVT     =       16.     ;CONVERT TO PHYSICAL ADDRESS XFC CODE
000021                          EXIT    =       17.     ;TERMINATE DM PROGRAM XFC CODE
                                ;
                                ;       GET STATUS OFFSETS
                                ;
000000                          ST.UNT  =       0.      ;UNIT NUMBER
000000                          ST.MSK  =       0.      ;SUBUNIT MASK
000001                          ST.STA  =       1.      ;STATUS BYTE
000001                          ST.MOD  =       1.      ;MODE BYTE
000002                          ST.ERR  =       2.      ;ERROR BYTE
000002                          ST.CON  =       2.      ;CONTROLLER BYTE
000002                          ST.C    =       2.      ;C BITS
000003                          ST.RTY  =       3.      ;RETRY COUNT/FAILURE CODE
                                ;
                                ;       STATUS BIT DEFINITIONS
                                ;
000200                          ST.OA   =       200     ; ONLINE TO ANOTHER (SET IF DRIVE UNAVAILABLE)
000100                          ST.RR   =       100     ; READJUSTMENT BIT (SET IF RECALIBRATION REQUIRED)
000040                          ST.DR   =       40      ; DIAGNOSTIC REQUEST (SET IF DIAGNOSTIC REQUESTED)
000020                          ST.SR   =       20      ; SPINDLE READY (SET IF SPINDLE READY)
000002                          ST.PS   =       2       ; PORT SWITCH (SET IF PORT SWITCH IN)
000001                          ST.RU   =       1       ; RUN/STOP SWITCH (SET IF RUN/STOP SWITCH IN)
000200                          ST.FE   =       200     ; FATAL ERROR (SET IF FATAL ERROR OCCURRED)
000100                          ST.RE   =       100     ; RETRIABLE ERROR (SET IF RETRIABLE ERROR OCCURRED)
000040                          ST.PE   =       40      ; PROTOCOL ERROR (SET IF PROTOCOL ERROR OCCURRED)
000020                          ST.DF   =       20      ; INITIALIZATION FAILURE (SET IF INIT FAILED)
000010                          ST.WE   =       10      ; WRITE ENABLE (SET IF WRT ATTEMPTED ON PROT DISK)
002000                          ST.FO   =       2000    ; FORMATTING (SET IF FORMATTING ENABLED)
001000                          ST.DB   =       1000    ; DIAGNOSTIC CYLS (SET IF DIAG CYL ACCESS ENABLED)
000400                          ST.S7   =       400     ; SECTOR SIZE (SET FOR 576 BYTE SECTORS)
```

```
                                ;       GET COMMON CHARACTERISTICS OFFSETS
                                ;
000000                          SHRTTO  =       0.              ;SHORT TIMEOUT <3:0>
000000                          SDIVER  =       0.              ;SDI VERSION <7:4>
000000                          XFERRT  =       0.              ;TRANSFER RATE <15:0>
000001                          LONGTO  =       1.              ;LONG TIMEOUT <3:0>
000001                          RETS    =       1.              ;RETRIES <7:4>
000001                          RCTCPS  =       1.              ;F/RCT COPIES <11:8>
000001                          SS      =       1.              ;SECTOR SIZE <15:15>
000002                          ERLEV   =       2.              ;ERROR RETRY LEVELS <7:0>
000002                          ECCRSH  =       2.              ;ECC THRESHOLD <15:8>
000003                          MICREV  =       3.              ;MICROCODE REVISION NUMBER <7:0>
000003                          HRDREV  =       3.              ;HARDWARE REVISION NUMBER <15:8>
000004                          DRVID   =       4.              ;UNIQUE DRIVE ID <47:0>
000007                          DRTYPE  =       7.              ;DRIVE TYPE IDENTIFIER <7:0>
000007                          REVS    =       7.              ;REVS/SECOND <15:8>
                                ;
                                ;       GET SUBUNIT CHARACTERISTICS OFFSETS
                                ;
                                ;THESE OFFSETS ARE CURRENTLY GIVEN AS FOLLOWING THE COMMON CHARACTERISTICS
                                ;
000013                          SUB     =       11.             ;OFFSET TO PUT SUBUNIT AFTER COMMON;
000000                          LBNCYL  =       0.              ;NUMBER OF CYLINDERS IN LBN AREA <31:0>
000001                          HICYL   =       1.              ;HI ORDER CYLINDER BITS <15:12>
000002                          GRPCYL  =       2.              ;GROUPS PER CYLINDER <7:0>
000002                          HILBN   =       2.              ;HI STARTING LBN <11:8>
000002                          HIXBN   =       2.              ;HI STARTING XBN <15:12>
000003                          TRKGRP  =       3.              ;TRACKS PER GROUP <7:0>
000003                          HIRBN   =       3.              ;HI STARTING RBN <11:8>
000003                          HIDBN   =       3.              ;HI STARTING DBN <15:12>
000004                          RBNTRK  =       4.              ;RBNS PER TRACK <6:0>
000004                          RM      =       4.              ;REMOVABLE MEDIA <7:7, 1=REMOVEABLE
000005                          DATPRE  =       5.              ;DATA PREAMBLE SIZE IN WORDS <7:0>
000005                          HDRPRE  =       5.              ;HEADER PREAMBLE SIZE IN WORDS <15:8>
000006                          MEDTYP  =       6.              ;MEDIA TYPE <31:0>
000010                          FCTSIZ  =       8.              ;FCT COPY SIZE <15:0>
000011                          LBNTRK  =       9.              ;LBNS PER TRACK <7:0>
000011                          GRPOFF  =       9.              ;GROUP OFFSET (SECTORS) <15:8>
000012                          LBNHST  =       10.             ;LBNS IN HOST AREA <31:0>
000014                          RCTCSZ  =       12.             ;RCT COPY SIZE <15:0>
000021                          XBNCYL  =       17.             ;CYLS IN XBN AREA <15:0>
000022                          DBNCYL  =       18.             ;CYLS IN DBN AREA <15:8>
                                ;
                                ;       UNIT CODES
                                ;
000001                          UNIT0   =       1.              ;UNIT ZERO CODE
000002                          UNIT1   =       2.              ;UNIT ONE CODE
000004                          UNIT2   =       4.              ;UNIT TWO CODE
000010                          UNIT3   =       8.              ;UNIT THREE CODE
                                ;
                                ;       BIT MASK DEFINITIONS
                                ;
                                ;
177400                          HIBYTE  =       177400          ;HIGH BYTE MASK
000377                          LOBYTE  =       000377          ;LOW BYTE MASK
007777                          HBHINB  =       7777            ;HI BYTE, HI NIBBLE MASK
170377                          HBLONB  =       170377          ;HI BYTE, LO NIBBLE MASK
```

```
     58          177417              LBHINB  =       177417          ;LO BYTE, HI NIBBLE MASK
     59          177760              LBLONB  =       177760          ;LO BYTE, LO NIBBLE MASK
     60                              ;
     61          000001              TIMEOUT =       1.              ;DRIVE TIMEOUT CODE
     62          000002              HEADER  =       2.              ;HEADER COMPARE FAILURE CODE
     63          000004              REVECT  =       4.              ;REVECTOR NEEDED CODE
     64                              ;
     65          000002              WRONG   =       2.              ;FIRST WORD NOT START FRAME CODE
     66          000004              FRAME   =       4.              ;FRAMING ERROR CODE
     67          000010              CHECK   =       8.              ;CHECKSUM ERROR CODE
     68                              ;
     69          000001              TOOBIG  =       1.              ;NUMBER OF WORDS EXCEEDS 7064
     70          000002              LOW     =       2.              ;DM BUFFER ADDRESS IS LESS THAN 714
     71                              ;
     72                              ;       BIT MASK
     73                              ;
     74          000001              LARGE   =       1.              ;BLOCK NUMBER TOO LARGE
     75          000002              OVERFL  =       2.              ;SECTOR NUMBER LARGER THAN 16 BITS
```

```
                        ;MAINTANENCE READ/WRITE REQUEST NUMBERS
                        ;
060000  T1MSIZ  =      0.+DU.SPC        ;GET FREE MEMORY PARAMETERS
060001  T2DLL   =      1.+DU.SPC        ;DOWNLINE LOAD DRIVE DIAGNOSTIC
060002  T2CMD   =      2.+DU.SPC        ;MANUAL INTERVENTION TEST 2 PROTOCOL
060003  T4MPRM  =      3.+DU.SPC        ;GET MASTER PARAMETERS FROM SW QUESTIONS
060004  T4UPRM  =      4.+DU.SPC        ;GET UNIT PARAMETERS FROM HW QUESTIONS
060005  T4BB1   =      5.+DU.SPC        ;GET BAD BLOCKS (1 THRU 14)
060006  T4BB2   =      6.+DU.SPC        ;GET REST OF BAD BLOCKS (15 AND 16)
060007  T4SOFT  =      7.+DU.SPC        ;ADD TO SOFT ERROR AND ECC COUNT
060010  T4SEEK  =      8.+DU.SPC        ;ADD 1 TO SEEK COUNT
060011  T4MXFR  =      9.+DU.SPC        ;ADD TO MEGABITS READ AND WRITTEN
060012  UTOTST  =      10.+DU.SPC       ;GET UNITS TO TEST
060013  ERRMES  =      11.+DU.SPC       ;PRINT ERROR MESSAGE
060014  ERRMC   =      12.+DU.SPC       ;TEST 4 ERROR REPORTING
060015  MESSAG  =      13.+DU.SPC       ;INFORMATION MESSAGE
060016  DONE    =      14.+DU.SPC       ;MARK DM PROGRAM AS NO LONGER RUNNING
                        ;
                        ;       OTHER BIT DEFINITIIONS
                        ;
000001  RCVRDY  =      1                ; RECIEVER READY 1 = READY
000002  ATTN    =      2                ; ATTENTION BIT FOR RETURN DRIVE SIGNALS XFC
000004  RCVERR  =      4                ; RECIEVER ERROR
000100  AVAIL   =      100              ; AVAILABLE 1 = AVAILABLE
000400  XMTERR  =      400              ; TRANSMIT ERROR
100000  RWRDY   =      100000           ; IF SET, UDA IS ABLE TO READ AND/OR WRITE TO DRIVE
                        ;
                        ;       SDI COMMANDS AND RESPONSES
                        ;
000204  DISCON  =      204              ; DISCONNECT DRIVE
000006  ERECOV  =      6                ; ERROR RECOVERY
000201  CHGMOD  =      201              ; CHANGE MODE
000213  DRVONL  =      213              ; DRIVE ONLINE
000014  DRVRUN  =      14               ; DRIVE RUN
000005  DRVCLR  =      5                ; DRIVE CLEAR OPCODE
000207  GETCHR  =      207              ; GET CHARACTERISTICS
000210  GETSUB  =      210              ; GET SUBUNIT CHARACTERISTICS
000011  GETSTA  =      11               ; GET STATUS
000216  IRECLB  =      216              ; RECALIBRATE
000012  INSEEK  =      12               ; INITIATE SEEK
000176  COMPLT  =      176              ; SUCCESSFUL COMPLETION
000175  UNSSUC  =      175              ; UNSUCCESSFUL COMPLETION
000170  CHRRES  =      170              ; GET CHARACTERISTICS RESPONSE
000167  SBCRES  =      167              ; GET SUBUNIT CHARACTERISTICS RESPONSE
000366  STSRES  =      366              ; GET STATUS RESPONSE
000350  ECHOC   =      350              ; DIAGNOSTIC ECHO COMMAND AND RESPONSE
                        ;
                        ;       ERROR CODES
                        ;
000000  FTLSYS  =            0          ; SYSTEM FATAL ERROR
040000  FTLDEV  =        40000          ; DEVICE FATAL
100000  ERHARD  =       100000          ; HARD ERROR
140000  ERSOFT  =       140000          ; SOFT ERROR
```

```
                        .SBTTL  TEST 4 SPECIFIC INFORMATION
                        ;
                        ;       TEST 4 SPECIFIC INFORMATION
                        ;
                        ;
                        ;       CONSTANTS
                        ;
000377          SCTWRD  =       255.            ; NUMBER OF WORDS IN SECTOR TO FILL
000105          INTEDC  =       69.             ; INITIAL EDC VALUE
000061          TLEN.U  =       U.LGRP+1        ; UNIT PARAMETER LENGTH
007717          FIRSTU  =       HIMEM-TLEN.U    ; LOCATION OF FIRST UNIT PARAMETER BLOCK
                        ;
                        ;       UNIT PARAMETER OFFSETS
                        ;
000000          U.NEXT  =       0.              ; POINTER TO NEXT UNIT (RING LINKED LIST)
000001          U.SUBP  =       U.NEXT+1        ; 4 WORDS OF SUBUNIT PARAMETER POINTERS
000005          U.TIMO  =       U.SUBP+4        ; AREA TO STORE VARIOUS TIMEOUT VALUES
000006          U.RWTO  =       U.TIMO+1        ; READ/WRITE TIMEOUT AREA
000007          U.SEEK  =       U.RWTO+1        ; NUMBER OF SEEKS ISSUED
000010          U.NFUN  =       U.SEEK+1        ; NEXT FUNCTION ADDRESS (FOR DEFERRED CALLS)
000011          U.PAT   =       U.NFUN+1        ; PATTERN NUMBER TO WRITE
000012          U.CCNT  =       U.PAT+1         ; CURRENT COUNT OF T/G LOOPS
000014          U.PCTG  =       U.CCNT+2        ; POINTER TO CURRENT TRACK OR GROUP
000015          U.CTRK  =       U.PCTG+1        ; TRACK COUNT FOR GROUP OPERATIONS
000016          U.NSEC  =       U.CTRK+1        ; NUMBER OF SECTORS R/W THIS TRY
000017          U.MSEC  =       U.NSEC+1        ; NUMBER OF SECTORS TO BE R/W
000020          U.TSEC  =       U.MSEC+1        ; NUMBER OF SECTORS TO BE R/W THIS OP
000021          U.CSEC  =       U.TSEC+1        ; COUNT OF SECTORS R/W SO FAR
000022          U.MASK  =       U.CSEC+1        ; UNIT MASK FOR XFC CALLS (0001 - 1000)
000023          U.WRIT  =       U.MASK+1        ; WRITE PROTECTION STATUS
000024          U.ELEV  =       U.WRIT+1        ; CURRENT ERROR RECOVERY LEVEL
000025          U.RTRY  =       U.ELEV+1        ; MAXIMUM NUMBER OF READ RETRIES
000026          U.MLEV  =       U.RTRY+1        ; MAXIMUM NUMBER OF ERROR RECOVERY LEVELS
000027          U.ECCT  =       U.MLEV+1        ; ECC THRESHOLD
000030          U.SDIS  =       U.ECCT+1        ; SDI SHORT TIMEOUT
000031          U.SDIL  =       U.SDIS+1        ; SDI LONG TIMEOUT
000032          U.WPRT  =       U.SDIL+1        ; MASK TO WRITE PROTECT READ-ONLY DRIVES
000033          U.PARM  =       U.WPRT+1        ; UNIT PARAMETER WORD
000034          U.SUBU  =       U.PARM+1        ; SUBUNIT OFFSET (0 - 3)
000035          U.MBN   =       U.SUBU+1        ; MASTER L/DBN
000037          U.CBN   =       U.MBN+2         ; CURRENT L/DBN FOR START OF CHAIN
000041          U.RBN   =       U.CBN+2         ; RBN TO BE READ/WRITTEN IF LBN REVECTORED
000043          U.COPY  =       U.RBN+2         ; NUMBER OF RCT COPIES ON EACH SUBUNIT
000044          U.CCOP  =       U.COPY+1        ; CURRENT RCT COPY THAT WE'RE WORKING ON
000045          U.RWER  =       U.CCOP+1        ; ERROR (IF ANY) ON THE LAST R/W
000046          U.RVER  =       U.RWER+1        ; ERROR THAT OCCURRED BEFOR THE REVECTOR OPERATION
000047          U.SNUM  =       U.RVER+1        ; 4 WORDS THAT HOLD THE SUBUNIT LOGICAL NUMBERS
000053          U.CCYL  =       U.SNUM+4        ; CURRENT CYLINDER
000055          U.CGRP  =       U.CCYL+2
000056          U.LCYL  =       U.CGRP+1        ; LAST CYLINDER SEEKED TO
000060          U.LGRP  =       U.LCYL+2        ; LAST GROUP SEEKED TO
                        ;
                        ;       SUBUNIT PARAMETER OFFSET
                        ;
000000          S.PARM  =       0.              ; SUBUNIT PARAMETER WORD
000001          S.SDCL  =       S.PARM+1        ; STARTING DIAGNOSTIC CYLINDER
000003          S.PAT   =       S.SDCL+2        ; PATTERN TO USE FOR WRITES
```

```
                                S.TRKL  =       S.PAT+1                 ; NUMBER OF SECTORS IN ONE TRACK
                                S.SCHR  =       S.TRKL+1                ; POINTER TO SUBUNIT CHARACTERISTICS
                                S.MEGR  =       S.SCHR+1                ; SECTORS READ (UP TO 245)
                                S.MEGW  =       S.MEGR+1                ; SECTORS WRITTEN (UP TO 245)
                                S.BADP  =       S.MEGW+1                ; POINTER TO BAD BLOCK AREA
                                S.BESS  =       S.BADP+1                ; START OF BEGIN/END SETS
                                ;
                                ;       IF TRACK/GROUP LIMITS ARE GIVEN, THE SUBUNIT PARAMETERS HAVE THE
                                ;       FOLLOWING FIELDS ADDED TO THEM
                                ;
                                S.MCNT  =       S.BESS                  ; MAXIMUM TRACK/GROUP COUNT
                                S.TGOF  =       S.MCNT+2                ; ORIGINAL TRACK/GROUP OFFSET
                                S.TGSS  =       S.TGOF+2                ; START OF TRACK/GROUP SE`S
```

TEST 4 SPECIFIC INFORMATION

```
                                ;
                                ;       DUMMY SDI CONTROL BLOCK OFFSETS
                                ;
000001                          D.LIMT  =       1                       ; DUMMY SDI SEARCH LIMIT
000002                          D.SCHR  =       2                       ; DUMMY POI ER TO SUBUNIT CHAR-5
                                ;
                                ;
                                ;       UNIT PARAMETER BITS (U.PARM(R5))
                                ;
100000                          DROP    =       100000                  ; DROP BIT (SET IF UNIT OR SUBUNIT DROPPED)
040000                          INITW   =       40000                   ; INITIAL WRITE (SET IF INITIAL WRITE IN PROG)
020000                          RESEEK  =       20000                   ; IF 1, INDICATES THAT A SEEK IS NECESSARY
010000                          DIREC   =       10000                   ; DIRECTION (SET IF          AL ACCESSES DECREASING)
004000                          NEWSUB  =       4000                    ; SET IF          AL SEEKS MOVED TO NEW SUBUNIT
002000                          SEKINP  =       2000                    ; SEEK IN PROGRESS - SET IF TRUE
001000                          FTIME   =       1000                    ; FIRST TIME FLAG - SET FOR INIT CODE
000400                          REVEC   =       400                     ; REVECTOR BIT (SET IF BLOCK REVECTORED)
000200                          RBNBN   =       200                     ; SEE IF WORKING ON RBN
000100                          REDWRT  =       100                     ; REDWRT (READ OR WRITE IN PROGRESS SET IF WRITE)
000040                          REVINP  =       40                      ; REVECTORING OPERATION IN PROGRESS
000020                          DATERR  =       20                      ; DATA ERROR IF SET
000004                          RETRY   =       4                       ; IF CLEAR, START RETRIES AT ZERO
000001                          RCLB    =       1                       ; RECALIBRATION BIT (SET IF RECALIBRATE JUST DONE)
                                ;
                                ;       SUBUNIT PARAMETER BITS (S.PARM(R4))
                                ;
020000                          DCYLS   =       20000                   ; DIAGNOSTIC CYLINDER FLAG (SET IF DBNS)
010000                          ECCCHK  =       10000                   ; 1 IF ECC CORRECTION ALLOWED   MASTER BITS
004000                          RONLY   =       4000                    ; READ ONLY (SET IF READ ONLY)
002000                          WONLY   =       2000                    ; WRITE ONLY (SET IF WRITE ONLY)
001000                          RTRIES  =       1000                    ; 1 IF RETRIES ALLOWED
000200                          ONLYCL  =       200                     ; SET IF ONLY CYLINDERS SPECIFIED
                                                                        ; USED DURING SETUP ONLY
000100                                          100                     ;       AL SEEK (START UP TESTING ONLY)
000040                          BEUSED  =       40                      ; BEGIN/END SETS (USED IF SET)
000020                          TRACKS  =       20                      ; TRACKS OR GROUPS (TRACK IF SET)
000010                          WCHECK  =       10                      ; WRITE CHECK BIT (IF SET, WRITE CHECK WILL BE DONE)
                                ;                                   WCHECK IS ALSO USED FOR THE UNIT PARAMETERS
000004                          WCHKAL  =       4                       ; SET IF WRITE CHECK ALWAYS TO BE DONE
000002                          DATCMP  =       2                       ; DATA COMPARE (SET IF DATA COMPARE TO BE DONE)
                                ;                                   DATCMP IS ALSO USED FOR THE UNIT PARAMETERS
000001                          DCMPAL  =       1                       ; SET IF DATA COMPARE ALWAYS TO BE DONE
```

MACRO DEFINITIONS

```
                                .SBTTL  MACRO DEFINITIONS
                                ;
                                ;       DIAGNOSTIC MACRO FOR TEST4 OVERLAYS
                                ;
                                        .MACRO  DIAG$$
                                        TST     $$DIAG+$DIAG$
                                        BEQ     .+6
                                        MOV     #60000,R0
                                        MOV     R0,@$$DIAG+$DIAG$
                                        BR      .+1
                        $DIAG$  =       $DIAG$ + 1
                                        .ENDM
```

MACRO DEFINITIONS

```
                        ;       MESSAGE CONTROL TABLE MACRO
                        ;
                                .MACRO MSG CMDBUF,CMDSZ,RPLBUF,RPLSZ,SUCCOM
                                .WORD CMDBUF                    ;ADDRESS OF COMMAND
                                .WORD CMDSZ                     ;SIZE OF COMMAND IN BYTES
                                .WORD RPLBUF                    ;ADDRESS OF REPLY
                                .WORD RPLSZ                     ;SIZE OF REPLY IN WORDS
                                .IF NB NUMBER
                                .WORD SUCCOM                    ; SUCCESSFUL COMPLETION CODE
                                .ENDC
                                .ENDM
```

MACRO DEFINITIONS

```
      1                                .MACRO  BCS     LAB..
      2                                                        BCC     .+2
      3                                                        BR      LAB..
      4                                .ENDM
```

```
                        ;       PUSH REGISTER MACRO
                        ;
                                .MACRO PUSH R9
                                .IRP X,<R9>
                                                                MOV X,-(SP)
                                .ENDR
                                .ENDM
                        ;
                        ;       POP REGISTER MACRO
                        ;
                                .MACRO POP R9
                                .IRP X,<R9>
                                                                MOV (SP)+,X
                                .ENDR
                                .ENDM
```

MACRO DEFINITIONS

```
                        ;ERROR MACROS
                        ;THESE MACROS ARE CALLED TO REPORT ERRORS TO THE HOST PROGRAM.
                        ;THE MACRO NAMES ARE : ERRSF, ERRDF, ERRHRD, ERRSFT. EACH RESULTS IN THE HOST
                        ;BEING REQUESTED TO REPORT THE ER  R.
                        ;ARGUMENTS:     1 (MS$) MESSAGE P  NTER
                        ;               2 (P1$) PARAMETER #1
                        ;               3 (P2$) PARAMETER #2
                        ;               4 (P3$) PARAMETER #3
                        ;               5 (P4$) PARAMETER #4
                        ;               6 (P5$) PARAMETER #5
                        ;               7 (P6$) PARAMETER #6
                        ;               8 (P7$) PARAMETER #7
                        ;               9 (P8$) PARAMETER #8
                        ;
                        ;THE MESSAGE POINTER MUST POINT TO AN ADDRESS IN THE OVERLAY 'MS'' IMMEDIATELY
                        ;FOLLOWING THE MAIN CODE. ANY ADDRESS MODE MAY BE USED (E.G. #MS1, @R2).
                        ;THE ADDRESS MUST CONTAIN AN ASCII FORMAT STRING TO DETERMINE THE MESSAGE
                        ;TO PRINT.
                        ;THE PARAMETER ARGUMENTS ARE OPTIONAL. THEY SHOULD BE SUPPLIED ONLY WHEN
                        ;THERE IS DATA TO BE PASSED TO THE HOST THAT WILL BE USED IN PRINTING THE
                        ;MESSAGE. THESE PARAMETER ARGUMENTS ARE THE ADDRESS OF DATA TO BE PASSED
                        ;USING ANY ADDRESSING MODE DESIRED.
                        ;ALL REGISTERS ARE RETURNED UNCHANGED. IT SHOULD BE NOTED THAT ARGUMENTS
                        ;CONTAINING SOMETHING OTHER THAN A REGISTER NAME (E.G. #100 OR MEMADR)
                        ;ASSEMBLE TO INSTRUCTIONS THAT SAVE AND RESTORE A REGISTER ON THE STACK.

                                .MACRO ERRSF MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$
                                .NARG ARGS$
                                .RADIX 10
                                ERROR$ FTLSYS,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,\ERRN
                                .ENDM

                                .MACRO ERRDF MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$
                                .NARG ARGS$
                                .RADIX 10
                                ERROR$ FTLDEV,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,\ERRN
                                .ENDM

                                .MACRO ERRHRD MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$
                                .NLIST
                                .NARG ARGS$
                                .RADIX 10
                                ERROR$ ERHARD,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,\ERRN
                                .LIST
                                .ENDM

                                .MACRO ERRSFT MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$
                                .NARG ARGS$
                                .RADIX 10
                                ERROR$ ERSOFT,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,\ERRN
                                .ENDM
```

MACRO DEFINITIONS

```
                        ;THE FOLLOWING MACRO ACTUALLY PROCESSES THE ERROR CALL TO THE HOST PROGRAM

                .MACRO ERROR$ ET$,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,ERRN$
                  .RADIX 8
                  PRMS=ARGS$-1
                  .IF LT,<PRMS>
                    .ERROR;NOT ENOUGH ARGUMENTS IN ERROR CALL
                  .ENDC
                  REGS$=-1
                  .IIF GE,<PRMS-8.>,PARGS. P8$
                  .IIF GE,<PRMS-7.>,PARGS. P7$
                  .IIF GE,<PRMS-6.>,PARGS. P6$
                  .IIF GE,<PRMS-5.>,PARGS. P5$
                  .IIF GE,<PRMS-4.>,PARGS. P4$
                  .IIF GE,<PRMS-3.>,PARGS. P3$
                  .IIF GE,<PRMS-2.>,PARGS. P2$
                  .IIF GE,<PRMS-1.>,PARGS. P1$
                  .IF GE REGS$
                    RSTR$ \REGS$
                  .ENDC
                  .RADIX 10
                  .LIST
                                                                        CALL RERROR     ;ERROR # ERRN$'.
                  .NLIST
                  .RADIX 8
                  .LIST
                                                                        .WORD ET$+ERRN
                                                                        .WORD <PRMS*10000>+MS$
                  .NLIST
                  ERRN=ERRN+1
                .ENDM

                .MACRO PARGS.,ADDR$
                  .NTYPE PTYPE$,ADDR$
                  .IF EQ,<PTYPE$&70>
                    .IIF EQ,<PTYPE$&7>-REGS$,RSTR$ \REGS$
                    .LIST
                                                                        MOV ADDR$,-(SP)
                    .NLIST
                  .IFF
                    .IF EQ,<PTYPE$&7>-1         ;PICK A REGISTER TO USE
                      REGU$=2                   ;SELECT R2 IF R1 IS USED IN PARAMETER FETCH
                    .IFF
                      REGU$=1                   ;OTHERWISE USE R1
                    .ENDC
                    .IF NE,<REGU$-REGS$>        ;IF REGISTER NOT ALREADY SAVED
                      .IF GE,REGS$
                        RSTR$ \REGS$            ;RESTORE CURRENT SAVED REGISTER
                      .ENDC
                      SAVR$ \REGU$              ;THEN SAVE SELECTED REGISTER
                    .ENDC
                    GETP$ \REGS$,ADDR$
                  .ENDC
                .ENDM
```

MACRO DEFINITIONS

```
                        .MACRO SAVR$ REGN
                          .LIST
                                                        MOV R'REGN,SAVREG
                          .NLIST
                          REGS$=REGN
                        .ENDM

                        .MACRO RSTR$ REGN
                          .LIST
                                                        MOV SAVREG,R'REGN
                          .NLIST
                          REGS$=-1
                        .ENDM

                        .MACRO GETP$ REGN,ADDR$
                          .LIST
                                                        MOV ADDR$,R'REGN
                                                        MOV R'REGN,-(SP)
                          .NLIST
                        .ENDM
```

```
                        ;
                        ;       PRIMARY ERROR REPORTING (TEST 4)
                        ;

                        .MACRO  SOFTER  NUM,ARGS
                        ERROR   #ERSOFT,NUM,<ARGS>
                                                        MOV     #ERRMES,R2
                                                        MOV     R2,OUT.RQ

                        .ENDM

                        .MACRO  REPSFT  SFTFLG,ECCFG
                        .IF     NB,SFTFLG
                                                        MOV     #1,OUT.02
                        .IFF
                                                        CLR     OUT.02
                        .ENDC
                        .IF     NB,ECCFG
                                                        MOV     #1,OUT.03
                        .IFF
                                                        CLR     OUT.03
                        .ENDC
                                                        PUSH    R0
                                                        MOV     #U.SNUM,R0
                                                        ADD     R5,R0
                                                        ADD     U.SUBU(R5),R0
                                                        MOV     (R0),OUT.01
                                                        MOV     #T4SOFT,R0
                                                        CALL    HOSTRQ
                                                        POP     R0

                        .ENDM

                        .MACRO  HARDER  NUM,ARGS
                        ERROR   #ERHARD,NUM,<ARGS>
                                                        MOV     #ERRMC,R2
                                                        MOV     R2,OUT.RQ

                        .ENDM

                        .MACRO  DEVFTL  NUM,ARGS
                        ERROR   #FTLDEV,NUM,<ARGS>
                                                        MOV     #ERRMC,R2
                                                        MOV     R2,OUT.RQ

                        .ENDM

                        .MACRO  SYSFTL  NUM,ARGS
                        ERROR   #FTLSYS,NUM,<ARGS>
                                                        MOV     #ERRMC,R2
                                                        MOV     R2,OUT.RQ

                        .ENDM

                        .MACRO  ERROR   TYPE,NUM,ARGS
                        .RADIX  10
                        NUMPTR  =       5
                        MOVMSG  #ER'NUM,4
                        .IF     NB,<ARGS>
                        .IRP    X,<ARGS>
                        MOVMSG  X,\NUMPTR
                        NUMPTR  =       NUMPTR + 1
                        .ENDR
```

```
                        .ENDC
                                                        MOV     #NUM,OUT.02
                                                        BIS     TYPE,OUT.02
                                                        MOV     #.,OUT.01

                        .RADIX
                        .ENDM

                        .MACRO  CERROR  STNUM,ARGS
                        .RADIX  10
                NUMPTR  =       STNUM
                        .IRP    X,<ARGS>
                        MOVMSG  X,\NUMPTR
                NUMPTR  =       NUMPTR + 1
                        .ENDR
                        .RADIX
                        .ENDM

                        .MACRO  ERRORC  ARGS
                        .RADIX  10
                        .IRP    X,<ARGS>
                        MOVMSG  X,\NUMPTR
                NUMPTR  =       NUMPTR + 1
                        .ENDR
                        .RADIX
                        .ENDM

                        .MACRO  MOVMSG  ARG,INDX

                        .IF     LT,INDX-10
                                                        MOV     ARG,OUT.0'INDX
                        .IFF
                                                        MOV     ARG,OUT.'INDX
                        .ENDC
                        .ENDM

                        .MACRO  ENDERR  POS
                        .IF     NB,POS
                NDERR   POS
                        .IFF
                NDERR   \NUMPTR
                        .ENDC
                        .ENDM

                        .MACRO  NDERR   POS
                        .IF     NE,POS
                                                        BIS     #POS,ERRPOS     ; SET THE POSITION
                        .IFF
                                                        CLR     ERRPOS          ; CLEAR THE POSITION
                        .ENDC
                        .ENDM
;
;       MESSAGE REPORTING MACRO
;
                        .MACRO  MSSG    NUM,ARGS
                        .RADIX  10
                NUMPTR  =       3
                MOVMSG  #MS'NUM,2
```

```
                                .IF     NB,<ARGS>
                                .IRP    X,<ARGS>
                                MOVMSG  X,\NUMPTR
                                NUMPTR  =       NUMPTR + 1
                                .ENDR
                                .ENDC
                                                        PUSH    <R0,R1>
                                                        MOV     #U.SNUM,R1
                                                        ADD     R5,R1
                                                        ADD     U.SUBU(R5),R1
                                                        MOV     (R1),OUT.01
                                                        MOV     #MESSAG,R0
                                                        CALL    HOSTRQ
                                                        POP     <R1,R0>
                                .RADIX
                                .ENDM
                                :
                                :
                                .MACRO  MSSGE   NUM,ARGS
                                .RADIX  10
                                NUMPTR  =       3
                                MOVMSG  #'NUM,2
                                .IF     NB,<ARGS>
                                .IRP    X,<ARGS>
                                MOVMSG  X,\NUMPTR
                                NUMPTR  =       NUMPTR + 1
                                .ENDR
                                .ENDC
                                                        PUSH    <R0,R1>
                                                        MOV     #MESSAG,R0
                                                        CALL    HOSTRQ
                                                        POP     <R1,R0>
                                .RADIX
                                .ENDM
```

MACRO DEFINITIONS

```
                        ;RETURN DRIVE STATUS MACRO WITH ERROR REPORTING

                                .MACRO  DSTAT,LAB$,E1,E2
                                .NLIST
                                .NLIST  MEB
                                .LIST   ME
                                .LIST
                                CALL    RDSTAT                  ; GET DRIVE STATUS
                                BIT     #10000,R1               ; SEE IF ANY ERRORS
                                BEQ     2$                      ; IF NO ERROR, BRANCH
                                BIT     #4000,R1                ; SEE IF XMIT ERROR
                                BEQ     1$                      ; IF SO, BRANCH
                                ERRHRD  E1                      ; REPORT INVALID STATUS ERROR
                                BR      LAB$                    ; BRANCH TO DONE
                        1$:     ERRHRD  E2                      ; REPORT XMIT ERROR
                                BR      LAB$                    ; BRANCH TO DONE
                        2$:
                                .NLIST
                                .NLIST  ME
                                .LIST   MEB
                                .LIST
                                .ENDM
```

MACRO DEFINITIONS

```
                        ;
                        ;       XOR THE CONTENTS OF TWO REGISTERS
                        ;
                        .MACRO  RXOR    REG1,REG2
                        MOV     REG2,-(SP)              ; SAVE REGISTER REG2
                        BIC     REG1,REG2               ; CLEAR COORESPONDING BITS IN REG2
                        BIC     (SP)+,REG1              ; CLEAR COORESPONDING BITS IN REG1
                        BIS     REG1,REG2               ; OR WHAT'S LEFT
                        .ENDM
```

```
                        ;       SDI INTERCHANGE WITH DRIVE WITH ERROR REPORTING
                        ;
                                .MACRO  TALKX   ERRLAB,E1,E2
                                .NLIST
                                .NLIST  MEB
                                .LIST   ME
                                .LIST
                                CALL    TALKER                  ; INITIATE SDI INTERCHANGE
                                TST     R3                      ; SEE IF ERROR OCCURRED
                                BEQ     12$                     ; IF NOT, BRANCH
                                BPL     11$                     ; IF SO, BRANCH
                                ERRHRD  E1;SEND COMMAND ERROR
                                BR      ERRLAB
                        11$:    ERRHRD  E2;RECEIVE COMMAND ERROR
                                BR      ERRLAB
                        12$:
                                .NLIST
                                .NLIST  ME
                                .LIST   MEB
                                .LIST
                                .ENDM
```

```
                                .SBTTL  START OF TEST CODE
                                ;THE FOLLOWING IS FOR DEBUG PURPOSES ONLY. CAN BE NOP OR BREAKPOINT.

 000714  114007                         CLR R0                          ;CHANGE TO BREAKPOINT FOR DEBUG

                                ;INITIALIZE STACK

 000715  104206  001373                 MOV #STACK,SP                   ;SET UP STACK POINTER
 000717  001374                         BR      START                   ; BRANCH OVER SUPPORT CODE
                                .SBTTL  RDSTAT - GET DRIVE'S REAL TIME DRIVE STATE
 000720                         RDSTAT:
                                ;
                                ;RETURN DRIVE STATUS
                                ;STATUS RETURNED IN DM REGISTER 1
                                ;
 000720                                 PUSH    <R3,R0>                 ; SAVE R3 AND R0
 000720  100463                                                                 MOV R3,-(SP)
 000721  100467                                                                 MOV R0,-(SP)
 000722  104203  000003                 MOV     #3,R3                   ; ALLOW ONLY 3 ERRORS
 000724  060007                 STATLP: XFC     STATUS                  ; GET DRIVE'S STATUS
 000725  103201  014000                 BIC     #14000,R1               ; CLEAR ERROR PASSING BITS
 000727  102201  000400                 BIT     #XMTERR,R1              ; CHECK XMIT ERRORS
 000731  010737                         BEQ     STATOK                  ; IF NO ERRORS, BRANCH
 000732  117403                         DEC     R3                      ; DECREMENT TRANSMIT ERROR COUNT
 000733  050724                         BNE     STATLP                  ; IF ERROR COUNT INCOMPLETE, BRANCH
 000734  104201  010000                 MOV     #10000,R1               ; FLAG AS TRANSMIT ERROR
 000736  000746                         BR      STATEX                  ; BRANCH
 000737  102201  000004         STATOK: BIT     #RCVERR,R1              ; RECIEVER ERRORS
 000741  050746                         BNE     STATEX                  ; IF VALID, BRANCH
 000742  117403                         DEC     R3                      ; DECREMENT ERROR COUNT
 000743  050724                         BNE     STATLP                  ; IF ERROR COUNT NON-ZERO, BRANCH
 000744  104201  014000                 MOV     #14000,R1               ; FLAG AS INVALID STATUS ERROR
 000746                         STATEX: POP     <R0,R3>                 ; RESTORE R0, R3
 000746  104267                                                                 MOV (SP)+,R0
 000747  104263                                                                 MOV (SP)+,R3
 000750  000000                         RETURN                          ; RETURN TO CALLING MODULE
```

```
      1                                 .SBTTL  HOSTRQ - HOST REQUEST - REPORT ERRORS, MEGABYTES TRANSFERRED, ETC.
      2 000751                          HOSTRQ:
      3                                 ;
      4                                 ;SEND REQUEST BUFFER TO HOST AND WAIT FOR RESPONSE.
      5                                 ;CLEAR ARGUMENT AREA OF OUT BUFFER IN PREPARATION
      6                                 ;FOR NEXT HOSTRQ CALL.
      7                                 ;
      8                                 ;INPUTS:
      9                                 ;       R0 - HOST REQUEST NUMBER
     10                                 ;       OUT BUFFER LOADED WITH DATA
     11
     12 000751                                  PUSH <R0,R1,R2>
        000751  100467                                                                  MOV R0,-(SP)
        000752  100461                                                                  MOV R1,-(SP)
        000753  100462                                                                  MOV R2,-(SP)
     13 000754  104070  001006                  MOV R0,OUT.RQ                ;STORE REQUEST NUMBER IN BUFFER
     14 000756  104207  001006          SNDAGN: MOV #OUT.RQ,R0               ;SEND BUFFER TO HOST
     15 000760  104201  000043                  MOV #BUFSIZ,R1
     16 000762  060016                          XFC MRD
     17 000763  115001                          TST R1                       ;CHECK FOR ERROR
     18 000764  050756                          BNE SNDAGN                   ;IF ERROR, TRY AGAIN
     19 000765  104207  001051                  MOV #IN.RQ,R0                ;WAIT FOR RESPONSE FROM HOST
     20 000767  104201  000043                  MOV #BUFSIZ,R1
     21 000771  060017                          XFC MWR
     22 000772  104207  001007                  MOV #OUT.01,R0               ;CLEAR ARGUMENT WORDS IN BUFFER
     23 000774  104201  000034                  MOV #OUT.29-OUT.01,R1
     24 000776  114002                          CLR R2
     25 000777  100272                  CLRBUF: MOV R2,(R0)+
     26 001000  117401                          DEC R1
     27 001001  030777                          BPL CLRBUF
     28 001002                                  POP <R2,R1,R0>
        001002  104262                                                                 MOV (SP)+,R2
        001003  104261                                                                 MOV (SP)+,R1
        001004  104267                                                                 MOV (SP)+,R0
     29 001005  000000                          RETURN
```

```
                                        ;STORAGE AREA FOR MAINTENANCE WRITE AND READ BUFFERS
                                        ;
                                        ;OUT BUFFER - DATA TO SEND TO HOST
                                        ;
001006  000000                          OUT.RQ: .WORD 0                 ;HOST REQUEST CODE
001007  000000                          OUT.01: .WORD 0                 ;DATA ARGUMENT  1
001010  000000                          OUT.02: .WORD 0                 ;DATA ARGUMENT  2
001011  000000                          OUT.03: .WORD 0                 ;DATA ARGUMENT  3
001012  000000                          OUT.04: .WORD 0                 ;DATA ARGUMENT  4
001013  000000                          OUT.05: .WORD 0                 ;DATA ARGUMENT  5
001014  000000                          OUT.06: .WORD 0                 ;DATA ARGUMENT  6
001015  000000                          OUT.07: .WORD 0                 ;DATA ARGUMENT  7
001016  000000                          OUT.08: .WORD 0                 ;DATA ARGUMENT  8
001017  000000                          OUT.09: .WORD 0                 ;DATA ARGUMENT  9
001020  000000                          OUT.10: .WORD 0                 ;DATA ARGUMENT 10
001021  000000                          OUT.11: .WORD 0                 ;DATA ARGUMENT 11
001022  000000                          OUT.12: .WORD 0                 ;DATA ARGUMENT 12
001023  000000                          OUT.13: .WORD 0                 ;DATA ARGUMENT 13
001024  000000                          OUT.14: .WORD 0                 ;DATA ARGUMENT 14
001025  000000                          OUT.15: .WORD 0                 ;DATA ARGUMENT 15
001026  000000                          OUT.16: .WORD 0                 ;DATA ARGUMENT 16
001027  000000                          OUT.17: .WORD 0                 ;DATA ARGUMENT 17
001030  000000                          OUT.18: .WORD 0                 ;DATA ARGUMENT 18
001031  000000                          OUT.19: .WORD 0                 ;DATA ARGUMENT 19
001032  000000                          OUT.20: .WORD 0                 ;DATA ARGUMENT 20
001033  000000                          OUT.21: .WORD 0                 ;DATA ARGUMENT 21
001034  000000                          OUT.22: .WORD 0                 ;DATA ARGUMENT 22
001035  000000                          OUT.23: .WORD 0                 ;DATA ARGUMENT 23
001036  000000                          OUT.24: .WORD 0                 ;DATA ARGUMENT 24
001037  000000                          OUT.25: .WORD 0                 ;DATA ARGUMENT 25
001040  000000                          OUT.26: .WORD 0                 ;DATA ARGUMENT 26
001041  000000                          OUT.27: .WORD 0                 ;DATA ARGUMENT 27
001042  000000                          OUT.28: .WORD 0                 ;DATA ARGUMENT 28
001043  000000                          OUT.29: .WORD 0                 ;DATA ARGUMENT 29
001044  000000                          OUT.30: .WORD 0                 ;DATA ARGUMENT 30
001045  000000                          OUT.31: .WORD 0                 ;DATA ARGUMENT 31
001046  000000                          OUT.32: .WORD 0                 ;DATA ARGUMENT 32
001047  000000                          OUT.33: .WORD 0                 ;DATA ARGUMENT 33
001050  000000                          OUT.34: .WORD 0                 ;DATA ARGUMENT 34

                                        ;IN BUFFER - DATA RECEIVED FROM HOST
                                        ;
001051  000000                          IN.RQ:  .WORD 0                 ;HOST REQUEST CODE (ECHO)
001052  000000                          IN.01:  .WORD 0                 ;DATA ARGUMENT  1
001053  000000                          IN.02:  .WORD 0                 ;DATA ARGUMENT  2
001054  000000                          IN.03:  .WORD 0                 ;DATA ARGUMENT  3
001055  000000                          IN.04:  .WORD 0                 ;DATA ARGUMENT  4
001056  000000                          IN.05:  .WORD 0                 ;DATA ARGUMENT  5
001057  000000                          IN.06:  .WORD 0                 ;DATA ARGUMENT  6
001060  000000                          IN.07:  .WORD 0                 ;DATA ARGUMENT  7
001061  000000                          IN.08:  .WORD 0                 ;DATA ARGUMENT  8
001062  000000                          IN.09:  .WORD 0                 ;DATA ARGUMENT  9
001063  000000                          IN.10:  .WORD 0                 ;DATA ARGUMENT 10
001064  000000                          IN.11:  .WORD 0                 ;DATA ARGUMENT 11
001065  000000                          IN.12:  .WORD 0                 ;DATA ARGUMENT 12
001066  000000                          IN.13:  .WORD 0                 ;DATA ARGUMENT 13
001067  000000                          IN.14:  .WORD 0                 ;DATA ARGUMENT 14
```

```
     58 001070  000000                IN.15:  .WORD 0                 ;DATA ARGUMENT 15
     59 001071  000000                IN.16:  .WORD 0                 ;DATA ARGUMENT 16
     60 001072  000000                IN.17:  .WORD 0                 ;DATA ARGUMENT 17
     61 001073  000000                IN.18:  .WORD 0                 ;DATA ARGUMENT 18
     62 001074  000000                IN.19:  .WORD 0                 ;DATA ARGUMENT 19
     63 001075  000000                IN.20:  .WORD 0                 ;DATA ARGUMENT 20
     64 001076  000000                IN.21:  .WORD 0                 ;DATA ARGUMENT 21
     65 001077  000000                IN.22:  .WORD 0                 ;DATA ARGUMENT 22
     66 001100  000000                IN.23:  .WORD 0                 ;DATA ARGUMENT 23
     67 001101  000000                IN.24:  .WORD 0                 ;DATA ARGUMENT 24
     68 001102  000000                IN.25:  .WORD 0                 ;DATA ARGUMENT 25
     69 001103  000000                IN.26:  .WORD 0                 ;DATA ARGUMENT 26
     70 001104  000000                IN.27:  .WORD 0                 ;DATA ARGUMENT 27
     71 001105  000000                IN.28:  .WORD 0                 ;DATA ARGUMENT 28
     72 001106  000000                IN.29:  .WORD 0                 ;DATA ARGUMENT 29
     73 001107  000000                IN.30:  .WORD 0                 ;DATA ARGUMENT 30
     74 001110  000000                IN.31:  .WORD 0                 ;DATA ARGUMENT 31
     75 001111  000000                IN.32:  .WORD 0                 ;DATA ARGUMENT 32
     76 001112  000000                IN.33:  .WORD 0                 ;DATA ARGUMENT 33
     77 001113  000000                IN.34:  .WORD 0                 ;DATA ARGUMENT 34
     78         000043                BUFSIZ  =       . - IN.RQ       ;SIZE OF BUFFER
```

```
                                ;
                                ;       TALKER SENDS AND RECEIVES AN SDI INTERCHANGE
                                ;
                                ;INPUTS:
                                ;       R2 - SDI INTERCONNECT
                                ;       R3 - POINTER TO COMMAND TABLE CONTAINING APPROPRIATE COMMAND
                                ;       SDILTO/SDISTO SDI LONG AND SHORT TIMEOUTS, RESPECTIVELY
                                ;OUTPUTS:
                                ;       R0 - RETURN OP CODE FROM UNIT
                                ;       R3 - ERROR CODE - 0 = NO ERROR, 1 = RECEIVE ERROR, 100001 = SEND ERROR
                                ;
 12 001114                      TALKER: PUSH    <R1,R4>         ; SAVE REGISTERS
    001114  100461                                                              MOV R1,-(SP)
    001115  100464                                                              MOV R4,-(SP)
 13 001116  104237                      MOV     (R3)+,R0        ; SET ADR OF SDI COMMAND BUFFER
 14 001117  104231                      MOV     (R3)+,R1        ; SET BUFFER LENGTH
 15 001120  060004                      XFC     SEND            ; SEND COMMAND
 16 001121  115001                      TST     R1              ; DID UNIT ACCEPT COMMAND
 17 001122  011126                      BEQ     TALK1A          ; IF SO, BRANCH
 18 001123  104203  100001              MOV     #100001,R3      ; FLAG AS SEND ERROR
 19 001125  001157                      BR      TALK2B          ; BRANCH TO EXIT
 20 001126  106203  001374      TALK1A: CMP     #LONG,R3        ; SEE IF LONG TIMEOUT TO BE USED
 21 001130  071134                      BMI     1$              ; IF SO, BRANCH
 22 001131  104304  001162              MOV     SDISTO,R4       ; SET UP SHORT TIMEOUT
 23 001133  001136                      BR      TALK1B          ; BRANCH
 24 001134  104304  001163      1$:     MOV     SDILTO,R4       ; SET UP LONG TIMEOUT
 25 001136  104137      TALK1B: MOV     (R3),R0         ; SET DATA BUFFER ADDRESS
 26 001137  104631  000001              MOV     1(R3),R1        ; SET BUFFER LENGTH
 27 001141  060005                      XFC     RCV             ; SEND RECEIVE SDI COMMAND
 28 001142  115001                      TST     R1              ; DID ERROR OCCUR
 29 001143  011156                      BEQ     TALK2A          ; IF NOT, BRANCH
 30 001144  106201  000001              CMP     #1,R1           ; SEE IF TIMEOUT
 31 001146  011151                      BEQ     1$              ; IF SO, BRANCH
 32 001147  104013                      MOV     R1,R3           ; MOVE ERROR TYPE TO R3 FOR REPORTING
 33 001150  001157                      BR      TALK2B          ; EXIT
 34 001151  117404      1$:     DEC     R4              ; DECREMENT TIMEOUT VALUE
 35 001152  051136                      BNE     TALK1B          ; IF NOT TIMEOUT, BRANCH
 36 001153  104203  000001              MOV     #1,R3           ; FLAG AS RECIEVE ERROR
 37 001155  001157                      BR      TALK2B          ; BRANCH TO EXIT
 38 001156  114003      TALK2A: CLR     R3              ; FLAG AS NO ERRORS
 39 001157              TALK2B: POP     <R4,R1>         ; RESTORE R4, R1
    001157  104264                                                              MOV (SP)+,R4
    001160  104261                                                              MOV (SP)+,R1
 40 001161  000000                      RETURN
 41
 42 001162  000012      SDISTO: .WORD   10.             ; SDI SHORT TIMEOUT
 43 001163  000024      SDILTO: .WORD   20.             ; SDI LONG TIMEOUT
```

HOSTRQ - HOST REQUEST - REPORT ERRORS, MEGABYTES TRANSFERRED,

```
                                ;MREAD
                                ;
                                ;READ ONE WORD FROM UNIBUS MEMORY
                                ;INPUTS:
                                ;       R5 - ADDRESS OF WORD TO READ (LOW 16 BITS)
                                ;       R4 -    ''           (HIGH 2 BITS)
                                ;OUTPUTS:
                                ;       R0 - DATA READ
                                ;
001164                          MREAD:  PUSH <R2,R3>                    ;SAVE SOME REGISTERS
001164  100462                                                                  MOV R2,-(SP)
001165  100463                                                                  MOV R3,-(SP)
001166  104057                          MOV R5,R0                       ;PUT UNIBUS ADDRESS IN R0 AND R1
001167  104041                          MOV R4,R1
001170  104202  000001                  MOV #1,R2                       ;TRANSFER ONE WORD
001172  104203  001222                  MOV #UDATA,R3                   ;LOCATION TO PUT DATA
001174  060013                          XFC UREAD                       ;DO THE READ
001175  104307  001222                  MOV UDATA,R0                    ;GET DATA READ
001177                                  POP <R3,R2>                     ;RESTORE OTHER REGISTERS
001177  104263                                                                  MOV (SP)+,R3
001200  104262                                                                  MOV (SP)+,R2
001201  000000                          RETURN
```

```
                                        ;MWRITE
                                        ;
                                        ;WRITE ONE WORD TO UNIBUS MEMORY
                                        ;INPUTS:
                                        ;       R5 - ADDRESS OF WORD TO WRITE (LOW 16 BITS)
                                        ;       R4 -      "          (HIGH 2 BITS)
                                        ;       R0 - DATA TO WRITE
                                        ;OUTPUTS:

001202                                  MWRITE: PUSH <R0,R2,R3>                 ;SAVE SOME REGISTERS
001202  100467                                                                             MOV R0,-(SP)
001203  100462                                                                             MOV R2,-(SP)
001204  100463                                                                             MOV R3,-(SP)
001205  104070  001222                          MOV R0,UDATA                    ;PUT DATA TO BE WRITTEN INTO BUFFER
001207  104057                                  MOV R5,R0                       ;PUT UNIBUS ADDRESS IN R0 AND R1
001210  104041                                  MOV R4,R1
001211  104202  000001                          MOV #1,R2                       ;TRANSFER ONE WORD
001213  104203  001222                          MOV #UDATA,R3                   ;ADDRESS OF DATA WORD
001215  060014                                  XFC UWRITE                      ;DO THE WRITE
001216                                          POP <R3,R2,R0>                  ;RESTORE THE REGISTERS
001216  104263                                                                             MOV (SP)+,R3
001217  104262                                                                             MOV (SP)+,R2
001220  104267                                                                             MOV (SP)+,R0
001221  000000                                  RETURN

001222  000000                  UDATA:  .WORD 0                                 ;DATA BUFFER FOR TRANSFERS TO AND FROM
                                                                                ;UNIBUS MEMORY
```

```
                                ;XOR
                                ;
                                ;PERFORM XOR LOGIC FUNCTION ON TWO REGISTERS
                                ;INPUTS:
                                ;       R1, R2 - DATA TO BE XOR'ED
                                ;OUTPUTS:
                                ;       R1 - UNCHANGED
                                ;       R2 - XOR OF TWO INPUTS
                                ;
001223                          XOR:    PUSH R3                         ;SAVE R3
001223  100463                                                                  MOV R3,-(SP)
001224  104013                          MOV R1,R3
001225  103023                          BIC R2,R3
001226  103012                          BIC R1,R2
001227  101032                          BIS R3,R2
001230                          POP R3                                  ;RESTORE R3
001230  104263                                                                  MOV (SP)+,R3
001231  115002                          TST R2                          ;SET CONDITION CODES
001232  000000                          RETURN
```

```
                                TO:
                                ;
                                ;       CALCULATE THE TIMEOUT IN 9SEC INTERVALS (SDI RECEIVE XFC TAKES
                                ;       9 SEC)
                                ;
001233  104201  000001                  MOV     #1,R1           ; SET UP LOG2 SHIFTER
001235  110201                  1$:     ROL     R1              ; DOUBLE THE TIMEOUT VALUE
001236  103201  000001                  BIC     #1,R1           ; CLEAR THE LOW BIT
001240  117407                          DEC     R0              ; DECREMENT COUNT
001241  051235                          BNE     1$              ; IF COUNT INCOMPLETE, BRANCH
001242  114007                          CLR     R0              ; CLEAR 9SEC COUNT
001243  115407                  2$:     INC     R0              ; INCREMENT 9 SEC COUNT
001244  107201  000011                  SUB     #9.,R1          ; SUBTRACT 9 SEC FROM TIMEOUT
001246  031243                          BPL     2$              ; IF MORE TIME TO GO, BRANCH
001247  000000                          RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                ;RERROR
                                ;
                                ;REPORT ERROR TO HOST PROGRAM
                                ;THIS ROUTINE IS CALLED BY THE ERROR MACROS:
                                ;ERRSF, ERRDF, ERRHRD AND ERRSFT
001250                          RERROR:
001250                                  PUSH R0                         ;SAVE ONE REGISTER
001250  100467                                                                  MOV R0,-(SP)
001251  104067                          MOV SP,R0                       ;GET STACK POINTER
001252                                  PUSH <R1,R2,R3,R4>              ;SAVE MORE REGISTERS
001252  100461                                                                  MOV R1,-(SP)
001253  100462                                                                  MOV R2,-(SP)
001254  100463                                                                  MOV R3,-(SP)
001255  100464                                                                  MOV R4,-(SP)
001256  115407                          INC R0                          ;CHANGE SAVED STACK POINTER TO POINT TO
                                                                        ; ADDRESS OF LOCATION AFTER CALL
001257  104271                          MOV (R0)+,R1                    ;GET RETURN PC
001260  104202  001007                  MOV #OUT.01,R2                  ;GET ADDRESS OF WHERE TO PUT DATA
001262  117401                          DEC R1                          ;REDUCE TO PC OF ERROR CALL
001263  100221                          MOV R1,(R2)+                    ;PUT PC IN OUT BUFFER
001264  115401                          INC R1                          ;GET BACK TO RETURN PC
001265  104213                          MOV (R1)+,R3                    ;GET ERROR NUMBER AND TYPE
001266  100223                          MOV R3,(R2)+                    ;PUT IN BUFFER
001267  104303  001331                  MOV LUNIT,R3                    ;PUT UNIT NUMBER IN BUFFER
001271  100223                          MOV R3,(R2)+
001272  104113                          MOV (R1),R3
001273  103203  170000                  BIC #^C007777,R3                ;GET MESSAGE POINTER
001275  100223                          MOV R3,(R2)+                    ;CLEAR OTHER BITS
001276  104214                          MOV (R1)+,R4                    ;PUT IN OUT BUFFER
                                                                        ;GET COUNT OF PARAMETERS
                                                                        ;R1 IS NOW POINTING TO INSTRUCTION AFTER ERROR CALL
001277  110704                          SWAB R4
001300  110604                          ROR R4                          ;EXTRACT NUMBER OF PARAMETERS FROM ERROR MACRO
001301  110604                          ROR R4
001302  110604                          ROR R4
001303  110604                          ROR R4
001304  103204  177760                  BIC #177760,R4
001306  104040  001327                  MOV R4,SPADJU+1                 ;SAVE FOR LATER ADJUSTMENT OF STACK POINTER
001310  011315                          BEQ RERRCA                      ;BRANCH IF NO PARAMETERS
001311  104273                  RERRPA: MOV (R0)+,R3                    ;GET PARAMETER
001312  100223                          MOV R3,(R2)+                    ;STORE PARAMETER IN OUT BUFFER
001313  117404                          DEC R4                          ;COUNT THE PARAMETERS
001314  051311                          BNE RERRPA                      ;GET NEXT IF MORE
001315  100471                  RERRCA: MOV R1,-(R0)                    ;PUT RETURN ADDRESS ON STACK
001316  104207  060013                  MOV #ERRMES,R0                  ;SEND ERROR PACKET TO HOST PROGRAM
001320  020751                          CALL HOSTRQ
001321                                  POP <R4,R3,R2,R1,R0>            ;RESTORE REGISTERS
001321  104264                                                                  MOV (SP)+,R4
001322  104263                                                                  MOV (SP)+,R3
001323  104262                                                                  MOV (SP)+,R2
001324  104261                                                                  MOV (SP)+,R1
001325  104267                                                                  MOV (SP)+,R0
001326  105206  000000          SPADJU: ADD #0,SP                       ;ADJUST STACK OVER PARAMETERS
                                                                        ;VALUE CHANGED ABOVE
001330  000000                          RETURN

001331  177777                  LUNIT:  .WORD -1                        ;LOGICAL UNIT NUMBER (-1 FOR NOT AVAILABLE)
```

```
     48 001332  000000                SAVREG: .WORD 0                         ;STORAGE FOR REGISTER AT CALL TIME
```

```
                                ;STACK AREA
                                
001333  123456                        .WORD 123456            ;END MARKER FOR STACK
001334                              .BLKW 31.               ;STACK
001373  123456                  STACK:  .WORD 123456            ;MARKER FOR STACK UNDERFLOW
```

```
                                .SBTTL  FREE MEMORY CHECK

001750                                  ERRN=1000.                      ;START ERROR NUMBERS FROM 1000.

                                ;ASK HOST WHERE FREE MEMORY IS AND TO FILL IT WITH AN ADDRESS PATTERN

001374                          LONG:                                   ;UNUSED LABEL THAT MUST BE DEFINED
001374  104207  060000        START:  MOV #T1MSIZ,R0                    ;GET REQUEST NUMBER
001376  020751                        CALL HOSTRQ                       ;ASK HOST
001377  104207  001052                MOV #IN.01,R0                     ;TRANSFER DATA FROM HOST
001401  104201  002416                MOV #FWADR,R1                     ; TO STORAGE
001403  104202  000010                MOV #8.,R2
001405  104273                  FMEMFL: MOV (R0)+,R3
001406  100213                        MOV R3,(R1)+
001407  117402                        DEC R2
001410  051405                        BNE FMEMFL

                                ;READ ALL OF SPECIFIED MEMORY AND CHECK FOR DATA SAME AS ADDRESS

001411  104305  002416                MOV FWADR,R5                      ;GET STARTING ADDRESS
001413  104304  002417                MOV FWADRH,R4
001415  021164                  ACHK1:  CALL MREAD                        ;READ FROM UNIBUS MEMORY
001416  115001                        TST     R1                        ;WAS IT OK?
001417  011440                        BEQ     ACHK2                     ;IF SO, CONTINUE
001420  117401                        DEC     R1                        ;IF R1 = 1, NON EXITENT MEMORY
001421  051430                        BNE     1$                        ; IF NOT, CONTINUE
001422                                ERRHRD  MS1000,R5,R4
001422  100464                                                                  MOV R4,-(SP)
001423  100465                                                                  MOV R5,-(SP)
001424  021250                                                                  CALL RERROR     ;ERROR # 1000.
001425  101750                                                                  .WORD ERHARD+ERRN
001426  020000                                                                  .WORD <PRMS*10000>+MS1000
001427  001451                          BR      ACHK3                   ;CONTINUE
001430                          1$:     ERRHRD  MS1001,R5,R4,R5,R0
001430  100467                                                                  MOV R0,-(SP)
001431  100465                                                                  MOV R5,-(SP)
001432  100464                                                                  MOV R4,-(SP)
001433  100465                                                                  MOV R5,-(SP)
001434  021250                                                                  CALL RERROR     ;ERROR # 1001.
001435  101751                                                                  .WORD ERHARD+ERRN
001436  040063                                                                  .WORD <PRMS*10000>+MS1001
001437  001451                          BR      ACHK3                   ;CONTINUE

                                ;COMPARE DATA READ WITH EXPECTED

001440  106075                  ACHK2:  CMP R0,R5                        ;COMPARE DATA READ WITH ADDRESS
001441  011451                          BEQ ACHK3                       ;BRANCH IF A MATCH
001442                                  ERRHRD MS1002,R5,R4,R5,R0 ;UNIBUS ADDRESSING ERROR - INCORRECT DATA READ
001442  100467                                                                  MOV R0,-(SP)
001443  100465                                                                  MOV R5,-(SP)
001444  100464                                                                  MOV R4,-(SP)
001445  100465                                                                  MOV R5,-(SP)
001446  021250                                                                  CALL RERROR     ;ERROR # 1002.
001447  101752                                                                  .WORD ERHARD+ERRN
001450  040170                                                                  .WORD <PRMS*10000>+MS1002

                                ;INCREMENT TO NEXT LOCATION UNTIL ALL TESTED
```

```
     39
     40 001451  106305  002420          ACHK3:  CMP LWADR,R5                    ;CHECK IF AT LAST ADDRESS
     41 001453  051457                          BNE ACHK4
     42 001454  106304  002421                  CMP LWADRH,R4
     43 001456  011464                          BEQ BCHK                        ;GO TO NEXT TEST IF SO
     44 001457  105205  000002          ACHK4:  ADD #2,R5                       ;INCREMENT TEST ADDRESS
     45 001461  041415                          BCC ACHK1
     46 001462  115404                          INC R4                          ;CARRY TO HIGH BITS
     47 001463  001415                          BR ACHK1                        ;LOOP IF STILL IN SPECIFIED MEMORY
```

FREE MEMORY CHECK

```
                                        ;CHECK EACH ADDRESS LINE BY LOOKING AT TWO LOCATIONS WITH ONLY THAT
                                        ;ADDRESS LINE DIFFERENT AND VERIFYING TWO LOCATIONS ARE ACTUALLY ACCESSED
                                        
001464  104307  002417    BCHK:   MOV FWADRH,R0           ;LOAD TEST ADDRESS HIGH BITS
001466  104070  002427            MOV R0,TADRH
001470  104203  000002            MOV #2,R3               ;GET STARTING ADDRESS BIT TO TEST
001472  104032            BCHK1:  MOV R3,R2               ;GET ADDRESS BIT TO TEST
001473  104301  002416            MOV FWADR,R1            ;GET FIRST ADDRESS OF FREE MEMORY
001475  021223                    CALL XOR                ;CHANGE JUST THE ONE BIT IN ADDRESS
001476  104020  002426            MOV R2,TADR             ;STORE TEST ADDRESS
001500  022273                    CALL BCHKM              ;TEST THE MEMORY ADDRESS
001501  105033                    ADD R3,R3               ;GET NEW ADDRESS BIT TO TEST
001502  051472                    BNE BCHK1               ;LOOP IF STILL AN ADDRESS BIT TO TEST
                                        
001503  104307  002416            MOV FWADR,R0            ;NOW MOVE TO HIGH TWO BITS
001505  104070  002426            MOV R0,TADR             ;COPY LOW BITS TO TEST ADDRESS
001507  104203  000001            MOV #1,R3               ;START BIT
001511  104032            BCHK2:  MOV R3,R2               ;GET TEST BIT
001512  104301  002417            MOV FWADRH,R1           ;GET FIRST ADDRESS
001514  021223                    CALL XOR                ;CHANGE ONLY THE ONE BIT
001515  104020  002427            MOV R2,TADRH            ;STORE AS TEST ADDRESS
001517  022273                    CALL BCHKM              ;TEST THE MEMORY ADDRESS
001520  105033                    ADD R3,R3               ;CHANGE TEST BIT
001521  102203  000004            BIT #4,R3               ;CHECK IF PAST TWO ADDRESS BITS
001523  011511                    BEQ BCHK2               ;REPEAT FOR OTHER BIT
```

```
                                        ;TRANSFER LARGE BUFFERS OF DATA TO AND FROM THE HOST MEMORY.
                                        ;
                                        ;THE HOST MEMORY WILL BE DIVIDED INTO SEVERAL BUFFERS. ALL BUFFERS WILL
                                        ;BE WRITTEN WITH PATTERN 0 THEN READ AND THE DATA COMPARED TO PATTERN 0.
                                        ;EACH BUFFER WILL THEN BE WRITTEN TO PATTERN 1 WITH A READ OF ALL BUFFERS
                                        ;AFTER EACH WRITE. WHEN ALL BUFFERS HAVE BEEN WRITTEN WITH PATTERN 1, THEN
                                        ;THE          REPEATS BY WRITING EACH BUFFER WITH PATTERN 2 AND THEN
                                        ;WITH PATTERN 3.
                                        ;
                                        ;DATA PATTERNS: (EACH IS A REPETITION OF THREE WORDS)
                                        ;0 - 111111     1 - 177400      2 - 155555      3 - 000377
                                        ;    044444         007760          133333          170017
                                        ;    022222         000377          066666          177400

                                        ;BREAK THE HOST MEMORY INTO BUFFERS

001524  104202  007774        CCHK:   MOV #7774,R2                    ;COMPUTE SIZE OF DATA
001526  107202  002460                SUB #FREE,R2                    ;  BUFFER IN M MEMORY
001530  105022                        ADD R2,R2                       ;CHANGE TO BYTE COUNT
                                ; *** FIND OUT IF WRITABLE HOST SPACE IS LESS THAN AVAILABLE DM SPACE
001531  104304  002420                MOV     LWADR,R4                ;R4 = LAST WRITABLE HOST BUFFER WORD LOCATION
001533  104303  002421                MOV     LWADRH,R3               ;R3 = EXTENDED ADDRESS BITS OF SAME
001535  107304  002416                SUB     FWADR,R4                ;R4 = WRITABLE BUFFER SIZE
001537  041541                        BCC     1$                      ;IF DIDN'T CROSS PAGE BOUNDARY, CONTINUE
001540  117403                        DEC     R3                      ;ELSE, DECREMENT EXTENDED ADDRESS BITS BY 1
001541  107303  002417        1$:     SUB     FWADRH,R3               ;R3 = EXTENDED ADDRESS OF WRITABLE BUFFER SIZE
001543  105203  000000                ADD     #0,R3                   ;CLEAR CARRY
                                ; *** DIVIDE WRITABLE REGION SIZE BY HALF
001545  110603                        ROR     R3                      ;ROTATE EXTENDED ADDRESS BITS INTO CARRY
001546  110604                        ROR     R4                      ;ROTATE EXTENDED ADDRESS IN & GET AREA/2
001547  102204  000001                BIT     #BIT00,R4               ;MAKE SURE THE BYTE COUNT IS EVEN
001551  011553                        BEQ     2$                      ;IF IT IS, CONTINUE
001552  117404                        DEC     R4                      ;ELSE, FORCE BYTE COUNT TO BE EVEN
001553  115003        2$:             TST     R3                      ;IS R3 = 0?
001554  051560                        BNE     3$                      ;IF NOT, WRITABLE AREA >>> AVAILABLE DM BUFFER SPACE
001555  106042                        CMP     R4,R2                   ;IS WRITABLE AREA < AVAILABLE DM BUFFER SPACE
001556  041560                        BCC     3$                      ;IF NOT, CONTINUE
001557  104042                        MOV     R4,R2                   ;ELSE, DM SPACE > HOST WRITABLE SPACE
                                                                        ; STORE IN R2 & CONTINUE
001560  104205  002460        3$:     MOV #FREE,R5                    ;GET ADDRESS OF FIRST TABLE ENTRY
001562  104304  002416                MOV FWADR,R4                    ;GET ADDRESS OF FIRST
001564  104303  002417                MOV FWADRH,R3                   ;  BUFFER IN HOST
001566  114001                        CLR R1                          ;INIT COUNT OF BUFFERS
001567  100254        4$:             MOV R4,(R5)+                    ;STORE HOST BUFFER ADDRESS
001570  100253                        MOV R3,(R5)+                    ;  IN TABE ENTRY
001571  107202  000004                SUB #4,R2                       ;REDUCE BUFFER SIZE BY TABLE ENTRY
001573  105024                        ADD R2,R4                       ;INCREADE HOST ADDRESS TO
001574  041576                        BCC 5$                          ;  NEXT BUFFER
001575  115403                        INC R3
001576  106303  002421        5$:     CMP LWADRH,R3                   ;CHECK IF BUFFER LARGER
001600  071610                        BMI 7$                          ;  THEN HOST MEMORY REMAINING
001601  051606                        BNE 6$
001602  106304  002420                CMP LWADR,R4
001604  041606                        BCC 6$
001605  001610                        BR 7$
001606  115401        6$:             INC R1                          ;COUNT THE TABLE ENTRY
001607  001567                        BR 4$                           ;GO BACK AND DO AGAIN
```

```
      58
      59 001610  104050  002433          7$:     MOV R5,DATBUF           ;SAVE ADDRESS OF DATA BUFFER
      60 001612  105207  000000                  ADD #0,R0               ;CLEAR CARRY
      61 001614  110602                          ROR R2                  ;SAVE SIZE OF DATA BUFFER
      62 001615  104020  002434                  MOV R2,SIZBUF           ;  IN WORDS
      63 001617  104010  002430                  MOV R1,BUFCNT           ;SAVE COUNT OF BUFFERS
```

```
                                        ;WRITE ALL BUFFERS TO PATTERN 0

001621  114007                          CLR R0                  ;LOAD PATTERN 0
001622  104070  002432                  MOV R0,CURPAT           ;  IN SAVE WORD
001624  104070  002431          8$:     MOV R0,CURBUF           ;SELECT BUFFER 0
001626  021666                          CALL WRITE              ;WRITE THE BUFFER
001627  104307  002431                  MOV CURBUF,R0           ;INCREMENT TO NEXT BUFFER
001631  115407                          INC R0
001632  106307  002430                  CMP BUFCNT,R0           ;WRITE ANOTHER IF
001634  051624                          BNE 8$                  ;  NOT AT LAST
001635  021742                          CALL READ               ;READ ALL HOST BUFFERS

001636  104307  002432          9$:     MOV CURPAT,R0           ;INCREMENT PATTERN NUMBER
001640  115407                          INC R0
001641  106207  000004                  CMP #4,R0               ;EXIT SUBTEST IF NO
001643  011662                          BEQ DONECD              ;  NO PATTERNS REMAINING
001644  104070  002432                  MOV R0,CURPAT

001646  114007                          CLR R0                  ;POINT TO BUFFER 0
001647  104070  002431          10$:    MOV R0,CURBUF
001651  021666                          CALL WRITE              ;WRITE A BUFFER
001652  021742                          CALL READ               ;READ ALL HOST BUFFERS
001653  104307  002431                  MOV CURBUF,R0           ;INCREMENT TO NEXT BUFFER
001655  115407                          INC R0
001656  106307  002430                  CMP BUFCNT,R0           ;CHECK IF AT LAST
001660  051647                          BNE 10$                 ;  WRITE NEXT BUFFER
001661  001636                          BR 9$                   ;  WRITE NEXT PATTERN
```

```
      1 001662  104207  060016          DONECD: MOV #DONE,R0                    ;END OF PROGRAM, TELL PDP-11
      2 001664  020751                          CALL HOSTRQ
      3 001665  001662                          BR DONECD                       ;KEEP SENDING IF RETURNED
```

```
                                ;FILL A DATA BUFFER BEGINNING AT ADDRESS IN DATBUF AND WHOSE SIZE
                                ;IS IN SIZBUF WITH A DATA PATTERN SPECIFIED BY CONTENTS OF CURPAT.
                                ;WRITE THIS BUFFER TO HOST STARTING AT ADDRESS IN TWO WORDS POINTED
                                ;TO BY CURBUF. THEN PLACE THE PATTERN NUMBER IN THE SECOND WORD OF THE
                                ;HOST ADDRESS

                                ;FILL BUFFER WITH DATA PATTERN

001666  104307  002432       WRITE:  MOV CURPAT,R0           ;GET PATTERN NUMBER
001670  105077                       ADD R0,R0               ;  TIMES FOUR
001671  105077                       ADD R0,R0
001672  104072                       MOV R0,R2               ;SAVE FOR ADDING TO TABLE ENTRY
001673  105207  002254               ADD #PAT0,R0            ;ADD START OF PATTERN TABLES
001675  104273                       MOV (R0)+,R3            ;GET PATTERN WORDS
001676  104274                       MOV (R0)+,R4
001677  104275                       MOV (R0)+,R5
001700  104307  002433               MOV DATBUF,R0           ;GET ADDRESS OF BUFFER
001702  104301  002434               MOV SIZBUF,R1           ;GET SIZE OF BUFFER

001704  100273                 1$:   MOV R3,(R0)+            ;LOAD ONE WORD
001705  117401                       DEC R1                  ;COUNT THE WORDS
001706  011715                       BEQ 2$
001707  100274                       MOV R4,(R0)+            ;LOAD ONE WORD
001710  117401                       DEC R1                  ;COUNT THE WORDS
001711  011715                       BEQ 2$
001712  100275                       MOV R5,(R0)+            ;LOAD ONE WORD
001713  117401                       DEC R1                  ;COUNT THE WORDS
001714  051704                       BNE 1$

                                ;WRITE THE BUFFER TO HOST MEMORY

001715  104305  002431           2$:   MOV CURBUF,R5           ;GET POINTER TO HOST ADDRESS
001717  105055                       ADD R5,R5               ;  COUNT TIMES TWO
001720  105205  002460               ADD #FREE,R5            ;  PLUS START ADDRESS
001722  104257                       MOV (R5)+,R0            ;GET LOW ADDRESS BITS
001723  104151                       MOV (R5),R1             ;GET HIGH ADDRESS BITS
001724  103201  177774               BIC #177774,R1          ;CLEAR CURRENT PATTERN
001726  104070  002447               MOV     R0,LBUFWL       ;SAVE LAST BUFFER WRITTEN
001730  104010  002450               MOV     R1,LBUFWH
001732  101012                       BIS R1,R2               ;SET IN NEW PATTERN
001733  100152                       MOV R2,(R5)             ;STORE IN TABLE ENTRY
001734  104302  002434               MOV SIZBUF,R2           ;GET WORDS TO TRANSFER
001736  104303  002433               MOV DATBUF,R3           ;GET DM ADDRESS
001740  060014                       XFC UWRITE              ;WRITE TO THE HOST MEMORY
001741  000000                       RETURN
```

```
                                ;READ AND PERFORM A DATA COMPARE ON ALL THE HOST BUFFERS. TWO WORD
                                ;TABLE ENTRIES STARTING AT FREE CONTAIN THE HOST ADDRESS OF THE
                                ;BUFFERS AND THE PATTERN NUMBER LAST WRITTEN TO THE BUFFER. THE
                                ;NUMBER OF BUFFERS IS IN BUFCNT AND THE SIZE OF THE BUFFERS IS IN
                                ;SIZBUF. THE DATA CAN BE READ INTO THE DM AT ADDRESS IN DATBUF.

001742  104205  002460    READ:   MOV #FREE,R5                    ;GET ADDRESS OF TABLE ENTRIES
001744  104304  002430            MOV BUFCNT,R4                   ;GET COUNT OF BUFFERS

                                ;READ DATA FROM HOST BUFFER INTO DM MEMORY

001746  104303  002433    1$:     MOV     DATBUF,R3               ;R3 = ADDRESS OF DATA BUFFER IN DM MEMORY
001750  104302  002434            MOV     SIZBUF,R2               ;SIZE OF BUFFER
001752  104257                    MOV     (R5)+,R0                ;GET BUFFER ADDRESS
001753  104151                    MOV     (R5),R1
001754  103201  177774            BIC     #177774,R1              ;PLEAR PATTERN NUMBER
001756  104070  002426            MOV     R0,TADR                 ;SAVE ADDRESS IN CASE OF ERROR
001760  104010  002427            MOV     R1,TADRH
001762  060013                    XFC     UREAD                   ;READ THE WORD
001763  115001                    TST     R1                      ;WAS IT READ PROPERLY?
001764  012030                    BEQ     3$                      ;IF SO, GO ON TO DO MORE
001765  117401                    DEC     R1                      ;WAS IT AN NXM ERROR?
001766  052010                    BNE     2$                      ;IF NOT, REPORT PARITY ERROR
001767                            ERRHRD  MS1003,TADR,TADRH,SIZBUF
001767  104010  001332                                                    MOV R1,SAVREG
001771  104301  002434                                                    MOV SIZBUF,R1
001773  100461                                                            MOV R1,-(SP)
001774  104301  002427                                                    MOV TADRH,R1
001776  100461                                                            MOV R1,-(SP)
001777  104301  002426                                                    MOV TADR,R1
002001  100461                                                            MOV R1,-(SP)
002002  104301  001332                                                    MOV SAVREG,R1
002004  021250                                                            CALL RERROR     ;ERROR # 1003.
002005  101753                                                            .WORD ERHARD+ERRN
002006  030332                                                            .WORD <PRMS*10000>+MS1003
002007  002030                    BR      3$                      ;EXIT
002010                    2$:     ERRHRD  MS1004,TADR,TADRH,SIZBUF
002010  104010  001332                                                    MOV R1,SAVREG
002012  104301  002434                                                    MOV SIZBUF,R1
002014  100461                                                            MOV R1,-(SP)
002015  104301  002427                                                    MOV TADRH,R1
002017  100461                                                            MOV R1,-(SP)
002020  104301  002426                                                    MOV TADR,R1
002022  100461                                                            MOV R1,-(SP)
002023  104301  001332                                                    MOV SAVREG,R1
002025  021250                                                            CALL RERROR     ;ERROR # 1004.
002026  101754                                                            .WORD ERHARD+ERRN
002027  030452                                                            .WORD <PRMS*10000>+MS1004

                                ; *** SET UP OUT BUFFER IN CASE OF ERROR

002030                  3$:     PUSH    <R5,R4>
002030  100465                                                            MOV R5,-(SP)
002031  100464                                                            MOV R4,-(SP)
002032  104205  002460            MOV #FREE,R5                    ;GET ADDRESS OF TABLE ENTRIES
002034  104304  002430            MOV BUFCNT,R4                   ;GET COUNT OF BUFFERS
002036  106204  000004            CMP     #4,R4                   ;IS R4 > 4?
```

```
                        BCC     4$                      ;IF NOT, SKIP NEXT INSTRUCTION
000004                  MOV     #4,R4                   ;IF SO, FORCE R4 = 4
                ; *** STORE IN OUTPUT BUFFER JUST IN CASE OF ERROR
001013          4$:     MOV     #OUT.05,R0              ;R0 -> OUTPUT BUFFER
002434                  MOV     SIZBUF,R1
                        MOV     R1,(R0)+                ;GIVE BUFFER SIZE
                        DEC     R4                      ;ADJUST R4
002451                  MOV     BTABLE(R4),R1           ;GIVE START OF ALL BUFFERS
                        MOV     R1,(R0)+
                        INC     R4                      ;RESTORE R4
                5$:     MOV     (R5)+,R1                ;STORE BUFFER LO ADDRESS
                        MOV     R1,(R0)+
                        MOV     (R5)+,R1                ;STORE BUFFER HI ADDRESS
177774                  BIC     #177774,R1
                        MOV     R1,(R0)+
                        DEC     R4                      ;DECREMENT (# OF BUFFERS)
                        BNE     5$                      ;IF NOT DONE, CONTINUE
002445                  MOV     R0,OUTPTR               ;OUT POINTER HAS ADDRESS PAST BUFFERS LOCATIONS
000006  002445          ADD     #6,OUTPTR               ;OUTPRT -> PAST DATA OF FIXED LENGTH
002446                  MOV     R0,OUTPT2               ;SAVE PREVIOUS POINTER
                        POP     <R4,R5>
104264                                                                  MOV (SP)+,R4
104265                                                                  MOV (SP)+,R5

                ;COMPARE DATA READ WITH DATA PATTERN

002434  002443          MOV     SIZBUF,TEMP1            ;TEMP1 IS DECREMENT COUNTER
002433                  MOV     DATBUF,R3               ;R3 -> DM DATA BUFFER
                        DEC     R5                      ;R5 -> BACK TO LO BUFFER ADDRESS
002426                  MOV     (R5)+,TADR              ;GET BUFFER ADDRESS
002427                  MOV     (R5),TADRH              ;
177774  002427          BIC     #177774,TADRH           ;MASK HI ADDRESS ONLY
                        MOV     (R5)+,R2                ;R2 = PATTERN
177763                  BIC     #177763,R2              ;MASK PATTERN
002254                  ADD     #PAT0,R2                ;R2 -> PATTERN
002442                  CLR     ERRNUM                  ;CLEAR ERROR COUNT
000003  002444  6$:     MOV     #3,TEMP2                ;TEMP2 = SIZE OF ANY PATTERN
                7$:     MOV     (R3)+,R0                ;DATA RECEIVED
                        MOV     (R2)+,R1                ;DATA EXPECTED
                        CMP     R0,R1                   ;COMPARE THE TWO
012157                  BEQ     8$                      ;IF EQUAL, NO ERROR

                ; *** ERROR HERE

002442                  INC     ERRNUM                  ;INCREASE THE ERROR COUNT
000003  002442          CMP     #3,ERRNUM               ;MORE THAN 3 ERROR REPORTS?
072157                  BMI     8$                      ;IF SO, DO NOT ADD ANY MORE TO THE ERROR REPORT
                        PUSH    <R0,R1>
100467                                                                  MOV R0,-(SP)
100461                                                                  MOV R1,-(SP)
002445                  MOV     OUTPTR,R0               ;R0 -> WHERE IN THE OUTPUT BUFFER
                                                        ; TO PUT ERROR INFO
002426                  MOV     TADR,R1
                        MOV     R1,(R0)+                ;STORE ERROR ADDRESS
002427                  MOV     TADRH,R1
                        MOV     R1,(R0)+                ;
                        MOV     (SP)+,R1                ;STORE DATA EXPECTED
```

```
     87 002152  100271                          MOV     R1,(R0)+
     88 002153  104261                          MOV     (SP)+,R1            ;STORE DATA RECEIVED
     89 002154  100271                          MOV     R1,(R0)+
     90 002155  104070  002445                  MOV     R0,OUTPTR           ;STORE NEW OUTPUT POINTER
     91
     92                                 ; *** RESUME NORMAL PROCESSING
     93
     94 002157  105200  000002  002426  8$:     ADD     #2,TADR             ;INCREMENT LO ADDRESS
     95 002162  052165                          BNE     9$                  ;IF DID NOT CROSS EXTENDED BOUNDARY, SKIP
     96 002163  115400  002427                  INC     TADRH               ;ELSE, INCREMENT EXTENDED ADDRESS
     97 002165  117400  002443          9$:     DEC     TEMP1               ;GONE OVER THE ENTIRE BUFFER?
     98 002167  012176                          BEQ     10$                 ;IF SO, EXIT LOOP
     99 002170  117400  002444                  DEC     TEMP2               ;AT END OF PATTERN?
    100 002172  052125                          BNE     7$                  ;IF NOT, LOOP BACK
    101 002173  107202  000003                  SUB     #3,R2               ;R2 -> START OF PATTERN
    102 002175  002122                          BR      6$                  ;LOOP BACK TO RESET TEMP2
    103
    104                                 ; *** ALL OF THE BUFFER HAS BEEN TESTED
    105
    106 002176  115000  002442          10$:    TST     ERRNUM              ;ANY ERRORS OCCURED?
    107 002200  012251                          BEQ     13$                 ;IF NOT, CHECK IF DONE WITH ROUTINE
    108 002201  104307  002446                  MOV     OUTPT2,R0           ;R0 -> AFTER BUFFER LOCATIONS STORED
    109 002203  104021                          MOV     R2,R1               ;R2 -> PATTERN
    110 002204  107201  002254                  SUB     #PAT0,R1            ;ISOLATE PATTERN #
    111 002206  103201  177763                  BIC     #177763,R1          ;MASK PATTERN NUMBER
    112 002210  105201  000000                  ADD     #0,R1               ;CLEAR CARRY
    113 002212  110601                          ROR     R1
    114 002213  110601                          ROR     R1                  ;PATTERN NUMBER IN LO 2 BITS
    115 002214  100271                          MOV     R1,(R0)+            ;STORE PATTERN NUMBER
    116 002215  104301  002432                  MOV     CURPAT,R1
    117 002217  100271                          MOV     R1,(R0)+            ;LAST PATTERN WRITTEN
    118 002220  104301  002447                  MOV     LBUFWL,R1
    119 002222  100271                          MOV     R1,(R0)+            ;LAST BUFFER WRITTEN
    120 002223  104301  002450                  MOV     LBUFWH,R1
    121 002225  100271                          MOV     R1,(R0)+
    122 002226  104301  002442                  MOV     ERRNUM,R1
    123 002230  100271                          MOV     R1,(R0)+            ;ERROR COUNT
    124 002231  106200  000003  002442          CMP     #3,ERRNUM           ;DO WE HAVE MORE THAN 3 ERRORS?
    125 002234  042240                          BCC     11$                 ;IF NOT, CONTINUE
    126 002235  104202  000002                  MOV     #2,R2               ;SET R2 FOR MAX
    127 002237  002243                          BR      12$                 ;
    128 002240  104302  002442          11$:    MOV     ERRNUM,R2           ;STORE ERROR COUNT IN R2
    129 002242  117402                          DEC     R2                  ;ADJUST R2
    130 002243  104621  002455          12$:    MOV     ETABLE(R2),R1       ;TABLE ENTRY
    131 002245  100171                          MOV     R1,(R0)
    132 002246                                  ERRHRD  MS1005
        002246  021250                                                      CALL RERROR     :ERROR # 1005.
        002247  101755                                                      .WORD ERHARD+ERRN
        002250  000560                                                      .WORD <PRMS*10000>+MS1005
    133
    134                                 ;CYCLE THROUGH ALL HOST BUFFERS
    135
    136 002251  117404          13$:    DEC     R4                  ;COUNT BUFFERS
    137 002252  051746                          BNE     1$                  ;REPEAT FOR ALL BUFFERS
    138 002253  000000          14$:    RETURN
    139
```

```
                                ;DATA PATTERN TABLES
                        
002254  111111          PAT0:   .WORD ^B1001001001001001
002255  044444                  .WORD ^B0100100100100100
002256  022222                  .WORD ^B0010010010010010
002257  000000                  .WORD 0
                        
002260  177400          PAT1:   .WORD ^B1111111100000000
002261  007760                  .WORD ^B0000111111110000
002262  000377                  .WORD ^B0000000011111111
002263  000000                  .WORD 0
                        
002264  155555          PAT2:   .WORD ^B1101101101101101
002265  133333                  .WORD ^B1011011011011011
002266  066666                  .WORD ^B0110110110110110
002267  000000                  .WORD 0
                        
002270  000377          PAT3:   .WORD ^B0000000011111111
002271  170017                  .WORD ^B1111000000001111
002272  177400                  .WORD ^B1111111100000000
```

FREE MEMORY CHECK

```
                                        ;CHECK IF TEST ADDRESS IS IN BOUNDS OF READABLE MEMORY

002273  106300  002427  002423  BCHKM:  CMP TADRH,FRADRH            ;COMPARE TEST ADDRESS WITH FIRST READABE ADDRESS
002276  012301                          BEQ 1$                      ;BRANCH IF EQUAL
002277  042306                          BCC 2$                      ;BRANCH IF TEST ADDRESS HIGHER
002300  002415                          BR BCHKMX                   ;BRANCH IF TEST ADDRESS LOWER
002301  106300  002426  002422  1$:     CMP TADR,FRADR
002304  042306                          BCC 2$                      ;BRANCH IF HIGHER OR SAME
002305  002415                          BR BCHKMX                   ;BRANCH IF LOWER
002306  106300  002425  002427  2$:     CMP LRADRH,TADRH            ;COMPARE TEST ADRESS WITH LAST READABLE ADDRESS
002311  012314                          BEQ 3$                      ;BRANCH IF EQUAL
002312  042321                          BCC 4$                      ;BRANCH IF READABLE ADDRESS HIGHER
002313  002415                          BR BCHKMX                   ;BRANCH IF READABLE ADDRESS LOWER
002314  106300  002424  002426  3$:     CMP LRADR,TADR
002317  042321                          BCC 4$                      ;BRANCH IF HIGHER OR SAME
002320  002415                          BR BCHKMX                   ;BRANCH IF LOWER

002321  104305  002416          4$:     MOV FWADR,R5                ;WRITE ONES INTO FIRST ADDRESS
002323  104304  002417                  MOV FWADRH,R4
002325  104207  177777                  MOV #177777,R0
002327  021202                          CALL MWRITE
002330  104305  002426                  MOV TADR,R5                 ;READ FROM TEST ADDRESS
002332  104304  002427                  MOV TADRH,R4
002334  021164                          CALL MREAD
002335  106201  000001                  CMP     #1,R1               ;DID WE GET A NXM?
002337  012415                          BEQ     BCHKMX              ;IF SO, EXIT
002340  106207  177777                  CMP #177777,R0              ;CHECK DATA READ FOR ALL ONES
002342  052415                          BNE BCHKMX                  ;GO TO NEXT BIT IF NOT ALL ONES

002343  104305  002416                  MOV FWADR,R5                ;WRITE ALL ZEROS TO FIRST ADDRESS
002345  104304  002417                  MOV FWADRH,R4
002347  114007                          CLR R0
002350  021202                          CALL MWRITE
002351  104305  002426                  MOV TADR,R5                 ;READ FROM TEST ADDRESS
002353  104304  002427                  MOV TADRH,R4
002355  021164                          CALL MREAD
002356  106201  000001                  CMP     #1,R1               ;DID WE GET A NXM?
002360  012415                          BEQ     BCHKMX              ;IF SO, EXIT
002361  115007                          TST R0                      ;CHECK DATA READ FOR ALL ZEROS
002362  052415                          BNE BCHKMX                  ;GO TO NEXT BIT IF NOT ALL ZEROS

002363  104301  002416                  MOV FWADR,R1                ;COMPUTE XOR OF FIRST ADDRESS
002365  104052                          MOV R5,R2                   ; AND TEST ADDRESS
002366  021223                          CALL XOR
002367  104027                          MOV R2,R0                   ;RESULT TO R0
002370  104301  002417                  MOV FWADRH,R1               ;NOW DO IT FOR HIGH BITS
002372  104042                          MOV R4,R2
002373  021223                          CALL XOR
002374                                  ERRHRD MS1006,FWADR,FWADRH,R5,R4,R0,R2 ;UNIBUS ADDRESSING ERROR. TWO ADDRESSES READ
002374  100462                                                                          MOV R2,-(SP)
002375  100467                                                                          MOV R0,-(SP)
002376  100464                                                                          MOV R4,-(SP)
002377  100465                                                                          MOV R5,-(SP)
002400  104010  001332                                                                  MOV R1,SAVREG
002402  104301  002417                                                                  MOV FWADRH,R1
002404  100461                                                                          MOV R1,-(SP)
002405  104301  002416                                                                  MOV FWADR,R1
```

```
        002407  100461
        002410  104301  001332                          MOV R1,-(SP)
        002412  021250                                  MOV SAVREG,R1
        002413  101756                                  CALL RERROR     ;ERROR # 1006.
        002414  061105                                  .WORD ERHARD+ERRN
     50                                                 .WORD <PRMS*10000>+MS1006
     51 002415  000000                  BCHKMX: RETURN
```

FREE MEMORY CHECK

```
                                        ;PROGRAM VARIABLES
                                        
002416  000000                  FWADR:  .WORD 0                         ;FIRST ADDRESS CONTAINING ADDRESS DATA
002417  000000                  FWADRH: .WORD 0
002420  000000                  LWADR:  .WORD 0                         ;LAST ADDRESS CONTAINING ADDRESS DATA
002421  000000                  LWADRH: .WORD 0
                                        
002422  000000                  FRADR:  .WORD 0                         ;FIRST ADDRESS READABLE
002423  000000                  FRADRH: .WORD 0
002424  000000                  LRADR:  .WORD 0                         ;LAST ADDRESS READABLE
002425  000000                  LRADRH: .WORD 0
                                        
002426  000000                  TADR:   .WORD 0                         ;TEST ADDRESS
002427  000000                  TADRH:  .WORD 0
                                        
002430  000000                  BUFCNT: .WORD 0                         ;COUNT OF BUFFERS IN HOST MEMORY
002431  000000                  CURBUF: .WORD 0                         ;CURRENT BUFFER BEING WRITTEN IN HOST
002432  000000                  CURPAT: .WORD 0                         ;CURRENT DATA PATTERN BEING WRITTEN
002433  000000                  DATBUF: .WORD 0                         ;ADDRESS OF DATA BUFFER IN DM MEMORY
002434  000000                  SIZBUF: .WORD 0                         ;SIZE OF DATA BUFFER IN HOST MEMORY
                                        
                                        ;HOST BUFFER TABLE ENTRIES ARE STORED AFTER THE PROGRAM AT FREE.
                                        ;       TWO WORDS PER TABLE
                                        ;               FIRST WORD  - LOW 16 BITS OF HOST ADDRESS
                                        ;               SECOND WORD - BITS 1-0 HIGH TWO BITS OF HOST ADDRESS
                                        ;                             BITS 3-2 PATTERN LAST WRITTEN
                                        
002435  000000                  CMPSIZ: .WORD 0                         ;TABLE FOR COMPARE DATA PATTERN XFC
002436  000000                  CMPADR: .WORD 0
002437  000000                  CMP1:   .WORD 0
002440  000000                  CMP2:   .WORD 0
002441  000000                  CMP3:   .WORD 0
                                        
                                        ;       MISSELENEOUS DATA STORAGE
002442  000000                  ERRNUM: .WORD 0                         ;HOLDS NUMBER OF ERRORS DURING COMPARE
002443  000000                  TEMP1:  .WORD 0                         ;TEMPORARY STORAGE
002444  000000                  TEMP2   .WORD 0                         ; SAME
002445  000000                  OUTPTR: .WORD 0                         ;POINTER INTO OUTPUT BUFFER FOR COMPARE
002446  000000                  OUTPT2: .WORD 0                         ; SAME
002447  000000                  LBUFWL: .WORD 0                         ;LAST BUFFER WRITTEN SAVED IN WRITE
002450  000000                  LBUFWH: .WORD 0                         ; SAME
002451  001274  001264  001254  BTABLE: .WORD B1,B2,B3,B4
002454  001244
002455  001357  001332  001305  ETABLE: .WORD E1,E2,E3
                                        ;MESSAGE STORAGE OVERLAY
                                        
002460                          FREE:                                   ; FREE SPACE BEGINS HERE
002460                                  DMOVLY MS,0
002460  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                        .NLIST BEX
                                        
000000    042    116    117    MS1000: .ASCII\'NON-EXISTANT MEMORY ERROR TRYING TO READ FROM UNIBUS.'N\
000034    123    062    065            .ASCII\S25''OCTAL''S6''HEX'N\
000045    123    070    042            .ASCII\S8''ADDRESS''S9018R9R9S5H18N\
000062    000                           .BYTE 0
000063    042    120    101    MS1001: .ASCII\'PARITY ERROR ON READ FROM UNIBUS.'N\
000105    123    062    065            .ASCII\S25''OCTAL''S6''HEX'N\
```

```
     56 000116      123      070      042            .ASCII\S8'ADDRESS'S9018R9R9S5H18N\
     57 000133      123      070      042            .ASCII\S8'DATA READ'S7016R9S5H16N\
     58 000150      123      070      042            .ASCII\S8'DATA EXPECTED'S3016R9S5H16N\
     59 000167      000                             .BYTE 0
     60 000170      042      125      116  MS1002:   .ASCII\'UNIBUS ADDRESSING ERROR - INCORRECT DATA READ.'N\
     61 000220      042      115      105            .ASCII\'MEMORY LOCATION SHOULD CONTAIN OWN ADDRESS.'N\
     62 000247      123      062      065            .ASCII\S25'OCTAL'S6'HEX'N\
     63 000260      123      070      042            .ASCII\S8'ADDRESS'S9018R9R9S5H18N\
     64 000275      123      070      042            .ASCII\S8'DATA READ'S7016R9S5H16N\
     65 000312      123      070      042            .ASCII\S8'DATA EXPECTED'S3016R9S5H16N\
     66 000331      000                             .BYTE 0
     67 000332      042      116      117  MS1003:   .ASCII\'NON-EXISTANT MEMORY ERROR TRYING TO READ FROM UNIBUS WITHIN BUFFER.'N\
     68 000375      123      062      070            .ASCII\S28'OCTAL'S6'HEX'N\
     69 000406      042      040      123            .ASCII\" STARTING ADDESS OF BUFFER '018R9R9S5H18N\
     70 000433      123      070      042            .ASCII\S8'BUFFER SIZE'S8016R9S5H16N\
     71 000451      000                             .BYTE 0
     72 000452      042      120      101  MS1004:   .ASCII\'PARITY ERROR ON READ FROM UNIBUS WITHIN BUFFER.'N\
     73 000503      123      062      070            .ASCII\S28'OCTAL'S6'HEX'N\
     74 000514      042      040      123            .ASCII\" STARTING ADDESS OF BUFFER '018R9R9S5H18N\
     75 000541      123      070      042            .ASCII\S8'BUFFER SIZE'S8016R9S5H16N\
     76 000557      000                             .BYTE 0
     77 000560      042      104      101  MS1005:   .ASCII\'DATA COMPARE FAILED AFTER WRITE THEN READ FROM UNIBUS.'N\
     78 000614      042      040      102            .ASCII\" BUFFER SIZE = '016''(D)'R9S3H16''(X)'R9S3D16''.(D)'N\
     79 000646      042      040      123            .ASCII\" STARTING ADDRESSES OF BUFFERS'N\
     80 000666      123      066      042            .ASCII\S6'OCTAL'S11'HEX'NR1\
     81 000700      042      040      103            .ASCII\" CURRENT DATA PATTERN READ'S17D6N\
     82 000721      042      040      114            .ASCII\" LAST PATTERN WRITTEN'S22D6N\
     83 000740      042      040      123            .ASCII\" STARTING ADDRESS OF LAST BUFFER WRITTEN'S3018''(O)'R9R9S3H18''(X)'N\
     84 001001      042      040      116            .ASCII\" NUMBER OF ERRORS FOUND'S20D16''.(D)'N\
     85 001024      123      064      042            .ASCII\S4'LOCATION'S6'DATA EXPECTED'S6'DATA RECEIVED'N\
     86 001054      123      062      042            .ASCII\S2'OCTAL   HEX'S6'OCTAL   HEX'S8'OCTAL   HEX'N\
     87 001103      122      061                     .ASCII\R1\
     88 001104      000                             .BYTE 0
     89 001105      042      125      116  MS1006:   .ASCII\'UNIBUS ADDRESSING ERROR. TWO ADDRESSES READ SAME LOCATION.'N\
     90 001143      123      063      063            .ASCII\S33'OCTAL'S6'HEX'N\
     91 001154      123      070      042            .ASCII\S8'KNOWN GOOD ADDRESS'S6018R9R9S5H18N\
     92 001177      123      070      042            .ASCII\S8'ERROR ADDRESS'S11018R9R9S5H18N\
     93 001217      123      070      042            .ASCII\S8'ADDRESS BIT IN ERROR'S4018R9R9S5H18N\
     94 001243      000                             .BYTE 0
     95
     96 001244      123      065      117  B4:       .ASCII\S5018R9R9S10H18N\
     97 001254      123      065      117  B3:       .ASCII\S5018R9R9S10H18N\
     98 001264      123      065      117  B2:       .ASCII\S5018R9R9S10H18N\
     99 001274      123      065      117  B1:       .ASCII\S5018R9R9S10H18N\
    100 001304      000                             .BYTE 0
    101
    102 001305      123      061      117  E3:       .ASCII\S1018S2R9R9H18S4016R9S2H16S7016R9S2H16S11N\
    103 001332      123      061      117  E2:       .ASCII\S1018S2R9R9H18S4016R9S2H16S7016R9S2H16S11N\
    104 001357      123      061      117  E1:       .ASCII\S1018S2R9R9H18S4016R9S2H16S7016R9S2H16S11N\
    105 001404      000                             .BYTE 0
    106
    107 001404                                      DMEND
        001405  000105                              .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
    108         000001                              .END
```

SYMBOL TABLE

```
ACHK1   001415        DRVID = 000004        HDRPRE= 000005        LARGE = 000001        OUT.21  001033
ACHK2   001440        DRVONL= 000213        HD.BAD= 110000        LBHINB= 177417        OUT.22  001034
ACHK3   001451        DRVRUN= 000014        HD.DBN= 140000        LBLONB= 177760        OUT.23  001035
ACHK4   001457        DU.DFL= 020000        HD.LBN= 000000        LBNCYL= 000000        OUT.24  001036
ARGS$ = 000007        DU.FTL= 050000        HD.PRV= 050000        LBNHST= 000012        OUT.25  001037
ATTN  = 000002        DU.INF= 030000        HD.RBN= 060000        LBNTRK= 000011        OUT.26  001040
AVAIL = 000100        DU.QUE= 010000        HD.REV= 030000        LBUFWH  002450        OUT.27  001041
BCHK    001464        DU.SPC= 060000        HD.XBN= 120000        LBUFWL  002447        OUT.28  001042
BCHKM   002273        DU.TER= 040000        HEADER= 000002        LINKLN= 000007        OUT.29  001043
BCHKMX  002415        D.LIMT= 000001        HIBYTE= 177400        LOBYTE= 000377        OUT.30  001044
BCHK1   001472        D.SCHR= 000002        HICYL = 000001        LONG    001374        OUT.31  001045
BCHK2   001511        ECC   = 000015        HIDBN = 000003        LONGTO= 000001        OUT.32  001046
BEUSED= 000040        ECCCHK= 010000        HILBN = 000002        LOW   = 000002        OUT.33  001047
BF.DAT= 000000        ECCFLG= 010000        HIMEM = 010000        LRADR   002424        OUT.34  001050
BF.ECC= 000401        ECCRSH= 000002        HIRBN = 000003        LRADRH  002425        OVERFL= 000002
BF.EDC= 000400        ECHO  = 000010        HIXBN = 000002        LUNIT   001331        OVE.MN= 000714
BIT00 = 000001        ECHOC = 000350        HOSTRQ  000751        LWADR   002420        OVE.MS= 000000
BIT01 = 000002        EOC   = 100000        HRDREV= 000003        LWADRH  002421        OVL.MN= 001545
BREAK = 000000        ERECOV= 000006        INITW = 040000        MEDTYP= 000006        OVL.MS= 001406
BTABLE  002451        ERHARD= 100000        INSEEK= 000012        MESSAG= 060015        OVL...= 003153
BUFCNT  002430        ERLEV = 000002        INTEDC= 000105        MICREV= 000003        OVSTRT= 007774
BUFFLG= 040000        ERRMC = 060014        IN.RQ   001051        MRD   = 000016        OVS.MN= 001040
BUFSIZ= 000043        ERRMES= 060013        IN.01   001052        MREAD   001164        OVS.MS= 004352
B1      001274        ERRN  = 001757        IN.02   001053        MS1000  000000        OV... = 003573
B2      001264        ERRNUM  002442        IN.03   001054        MS1001  000063        PAT0    002254
B3      001254        ERSOFT= 140000        IN.04   001055        MS1002  000170        PAT1    002260
B4      001244        ETABLE  002455        IN.05   001056        MS1003  000332        PAT2    002264
CCHK    001524        EXIT  = 000021        IN.06   001057        MS1004  000452        PAT3    002270
CHECK = 000010        E1      001357        IN.07   001060        MS1005  000560        PHYSA = 001000
CHGMOD= 000201        E2      001332        IN.08   001061        MS1006  001105        PRMS  = 000006
CHRRES= 000170        E3      001305        IN.09   001062        MWR   = 000017        PTYPE$= 000030
CLRBUF  000777        FB.DAT= 000000        IN.10   001063        MWRITE  001202        RBNBN = 000200
CMPADR  002436        FB.EDC= 000400        IN.11   001064        NEWSUB= 004000        RBNTRK= 000004
CMPSIZ  002435        FCTSIZ= 000010        IN.12   001065        ONLYCL= 000200        RBUFLN= 000415
CMP1    002437        FIRSTU= 007717        IN.13   001066        OUTPTR  002445        RCLB  = 000001
CMP2    002440        FMEMFL  001405        IN.14   001067        OUTPT2  002446        RCONT = 000000
CMP3    002441        FORMAT= 000001        IN.15   001070        OUT.RQ  001006        RCTCPS= 000001
COMPAR= 000006        FRADR   002422        IN.16   001071        OUT.01  001007        RCTCSZ= 000014
COMPLT= 000176        FRADRH  002423        IN.17   001072        OUT.02  001010        RCV   = 000005
CURBUF  002431        FRAME = 000004        IN.18   001073        OUT.03  001011        RCVERR= 000004
CURPAT  002432        FREE    002460        IN.19   001074        OUT.04  001012        RCVRDY= 000001
CVT   = 000020        FSTOP = 100000        IN.20   001075        OUT.05  001013        RDSTAT  000720
DATBUF  002433        FTIME = 001000        IN.21   001076        OUT.06  001014        READ    001742
DATCMP= 000002        FTLDEV= 040000        IN.22   001077        OUT.07  001015        REDWRT= 000100
DATERR= 000020        FTLSYS= 000000        IN.23   001100        OUT.08  001016        REGS$ = 177777
DATPRE= 000005        FT.BUF= 000000        IN.24   001101        OUT.09  001017        REGU$ = 000001
DBNCYL= 000022        FT.HI = 000001        IN.25   001102        OUT.10  001020        RERRCA  001315
DCMPAL= 000001        FT.LOW= 000002        IN.26   001103        OUT.11  001021        RERROR  001250
DCYLS = 020000        FWADR   002416        IN.27   001104        OUT.12  001022        RERRPA  001311
DINIT = 000011        FWADRH  002417        IN.28   001105        OUT.13  001023        RESEEK= 020000
DIREC = 010000        GETCHR= 000207        IN.29   001106        OUT.14  001024        RETRY = 000004
DISCON= 000204        GETSTA= 000011        IN.30   001107        OUT.15  001025        RETS  = 000001
DONE  = 060016        GETSUB= 000210        IN.31   001110        OUT.16  001026        REVEC = 000400
DONECD  001662        GRPCYL= 000002        IN.32   001111        OUT.17  001027        REVECT= 000004
DROP  = 100000        GRPOFF= 000011        IN.33   001112        OUT.18  001030        REVINP= 000040
DRTYPE= 000007        HBHINB= 007777        IN.34   001113        OUT.19  001031        REVS  = 000007
DRVCLR= 000005        HBLONB= 170377        IRECLB= 000216        OUT.20  001032        RM    = 000004
```

```
UDAT1 UNIBUS ADDRESSING DMACR X04.01 13-APR-82 15:48:42 PAGE 38-3
SYMBOL TABLE

RONLY = 004000          STSRES= 000366          S.SDCL= 000001          UDATA   001222          U.RBN = 000041
RREAL = 013400          ST.C  = 000002          S.TGOF= 000013          UNIT0 = 000001          U.RTRY= 000025
RSTOP = 100000          ST.CON= 000002          S.TGSS= 000015          UNIT1 = 000002          U.RVER= 000046
RTRIES= 001000          ST.DB = 001000          S.TRKL= 000004          UNIT2 = 000004          U.RWER= 000045
RWRDY = 100000          ST.DF = 000020          TADR    002426          UNIT3 = 000010          U.RWTO= 000006
RW.ANG= 000006          ST.DR = 000040          TADRH   002427          UNSSUC= 000175          U.SDIL= 000031
RW.BUF= 000001          ST.ERR= 000002          TALKER  001114          UREAD = 000013          U.SDIS= 000030
RW.CMD= 000004          ST.FE = 000200          TALK1A  001126          UTOTST= 060012          U.SEEK= 000007
RW.HI = 000003          ST.FO = 002000          TALK1B  001136          UWRITE= 000014          U.SNUM= 000047
RW.LOW= 000002          ST.MOD= 000001          TALK2A  001156          U.CBN = 000037          U.SUBP= 000001
RW.SDI= 000005          ST.MSK= 000000          TALK2B  001157          U.CCNT= 000012          U.SUBU= 000034
RW.STA= 000000          ST.OA = 000200          TEMP1   002443          U.CCOP= 000044          U.TIMO= 000005
SAVREG  001332          ST.PE = 000040          TEMP2   002444          U.CCYL= 000053          U.TSEC= 000020
SBCRES= 000167          ST.PS = 000002          TEST4 = 000000          U.CGRP= 000055          U.WPRT= 000032
SCTWRD= 000377          ST.RE = 000100          TIMEOU= 000001          U.COPY= 000043          U.WRIT= 000023
SDILTO  001163          ST.RR = 000100          TLEN.U= 000061          U.CSEC= 000021          WAITSI= 000012
SDISTO  001162          ST.RTY= 000003          TO      001233          U.CTRK= 000015          WBUFLN= 000401
SDIVER= 000000          ST.RU = 000001          TOOBIG= 000001          U.ECCT= 000027          WCHECK= 000010
SEKINP= 002000          ST.SR = 000020          TRACKS= 000020          U.ELEV= 000024          WCHKAL= 000004
SEND  = 000004          ST.STA= 000001          TRKGRP= 000003          U.LCYL= 000056          WCONT = 040000
        000100          ST.S7 = 000400          T1MSIZ= 060000          U.LGRP= 000060          WONLY = 002000
SHRTTO= 000000          ST.UNT= 000000          T2CMD = 060002          U.MASK= 000022          WREAL = 122400
SIZBUF  002434          ST.WE = 000010          T2DLL = 060001          U.MBN = 000035          WRITE   001666
SNDAGN  000756          SUB   = 000013          T4BB1 = 060005          U.MLEV= 000026          WRONG = 000002
SPADJU  001326          S.BADP= 000010          T4BB2 = 060006          U.MSEC= 000017          WSTOP = 140000
SS    = 000001          S.BESS= 000011          T4MPRM= 060003          U.NEXT= 000000          XBNCYL= 000021
STACK   001373          S.MCNT= 000011          T4MXFR= 060011          U.NFUN= 000010          XFERRT= 000000
START   001374          S.MEGR= 000006          T4SEEK= 060010          U.NSEC= 000016          XMTERR= 000400
STATEX  000746          S.MEGW= 000007          T4SOFT= 060007          U.PARM= 000033          XOR     001223
STATLP  000724          S.PARM= 000000          T4UPRM= 060004          U.PAT = 000011          XREAD = 000002
STATOK  000737          S.PAT = 000003          UDADM1= 001000 G        U.PCTG= 000014          XWRITE= 000003
STATUS= 000007          S.SCHR= 000005

. ABS.  007366      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  4477 WORDS  ( 18 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  71 PAGES
,B:UDAT1/C=$DMACRO,B:UDAT1
```

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

```
IN.34   21-77#
IN.RQ   20-19    21-43#   21-78
INITW    7-11#
INSEEK   5-40#
INTEDC   6-9#
IRECLB   5-39#
LARGE    4-74#
LBHINB   4-58#
LBLONB   4-59#
LBNCYL   4-23#
LBNHST   4-39#
LBNTRK   4-37#
LBUFWH  34-39*   35-120   38-41#
LBUFWL  34-38*   35-118   38-40#
LINKLN   2-121#
LOBYTE   4-55#
LONG    22-20    29-7#
LONGTO   4-6#
LOW      4-70#
LRADR   37-14    38-10#
LRADRH  37-10    38-11#
LUNIT   27-19    27-46#
LWADR   29-40    31-21    31-53    38-5#
LWADRH  29-42    31-22    31-50    38-6#
MEDTYP   4-35#
MESSAG   5-16#
MICREV   4-12#
MRD      3-17#   20-16
MREAD   23-10#   29-22    37-24    37-36
MS1000  29-27    38-50#
MS1001  29-29    38-54#
MS1002  29-36    38-60#
MS1003  35-24    38-67#
MS1004  35-26    38-72#
MS1005  35-132   38-77#
MS1006  37-49    38-89#
MWR      3-18#   20-21
MWRITE  24-10#   37-21    37-33
NEWSUB   7-14#
ONLYCL   7-32#
OUT.01  20-22    20-23    21-6#    27-13
OUT.02  21-7#
OUT.03  21-8#
OUT.04  21-9#
OUT.05  21-10#   35-37
OUT.06  21-11#
OUT.07  21-12#
OUT.08  21-13#
OUT.09  21-14#
OUT.10  21-15#
OUT.11  21-16#
OUT.12  21-17#
OUT.13  21-18#
OUT.14  21-19#
OUT.15  21-20#
OUT.16  21-21#
```

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

```
SNDAGN  20-14#   20-18
SPADJU  27-32*   27-42#
SS       4-9#
ST.C     3-30#
ST.CON   3-29#
ST.DB    3-47#
ST.DF    3-44#
ST.DR    3-37#
ST.ERR   3-28#
ST.FE    3-41#
ST.FO    3-46#
ST.MOD   3-27#
ST.MSK   3-25#
ST.OA    3-35#
ST.PE    3-43#
ST.PS    3-39#
ST.RE    3-42#
ST.RR    3-36#
ST.RTY   3-31#
ST.RU    3-40#
ST.S7    3-48#
ST.SR    3-38#
ST.STA   3-26#
ST.UNT   3-24#
ST.WE    3-45#
STACK   19-8     28-5#
START   19-9     29-8#
STATEX  19-25    19-27    19-31#
STATLP  19-18#   19-23    19-29
STATOK  19-21    19-26#
STATUS   3-10#   19-18
STSRES   5-45#
SUB      4-22#
T1MSIZ   5-3#    29-8
T2CMD    5-5#
T2DLL    5-4#
T4BB1    5-8#
T4BB2    5-9#
T4MPRM   5-6#
T4MXFR   5-12#
T4SEEK   5-11#
T4SOFT   5-10#
T4UPRM   5-7#
TADR    30-10*   30-16*   35-17*   35-24    35-24    35-26    35-26    35-61*   35-82    35-94*   37-7     37-14*   37-22    37-34
        38-13#
TADRH   30-5*    30-21*   35-18*   35-24    35-24    35-26    35-26    35-62*   35-63*   35-84    35-96*   37-3     37-10*   37-23
        37-35    38-14#
TALK1A  22-17    22-20#
TALK1B  22-23    22-25#   22-35
TALK2A  22-29    22-38#
TALK2B  22-19    22-33    22-37    22-39#
TALKER  22-12#
TEMP1   35-58*   35-97*   38-36#
TEMP2   35-68*   35-99*   38-37#
TEST4    2-38#
TIMEOU   4-61#
```

CROSS REFERENCE TABLE (CREF V04.00)

```
TLEN.U   6-10#    6-11
TO      26-1#
TOOBIG   4-69#
TRACKS   7-36#
TRKGRP   4-28#
U.CBN    6-41#    6-42
U.CCNT   6-22#    6-23
U.CCOP   6-44#    6-45
U.CCYL   6-48#    6-49
U.CGRP   6-49#    6-50
U.COPY   6-43#    6-44
U.CSEC   6-28#    6-29
U.CTRK   6-24#    6-25
U.ECCT   6-34#    6-35
U.ELEV   6-31#    6-32
U.LCYL   6-50#    6-51
U.LGRP   6-10     6-51#
U.MASK   6-29#    6-30
U.MBN    6-40#    6-41
U.MLEV   6-33#    6-34
U.MSEC   6-26#    6-27
U.NEXT   6-15#    6-16
U.NFUN   6-20#    6-21
U.NSEC   6-25#    6-26
U.PARM   6-38#    6-39
U.PAT    6-21#    6-22
U.PCTG   6-23#    6-24
U.RBN    6-42#    6-43
U.RTRY   6-32#    6-33
U.RVER   6-46#    6-47
U.RWER   6-45#    6-46
U.RWTO   6-18#    6-19
U.SDIL   6-36#    6-37
U.SDIS   6-35#    6-36
U.SEEK   6-19#    6-20
U.SNUM   6-47#    6-48
U.SUBP   6-16#    6-17
U.SUBU   6-39#    6-40
U.TIMO   6-17#    6-18
U.TSEC   6-27#    6-28
U.WPRT   6-37#    6-38
U.WRIT   6-30#    6-31
UDADM1   2-40#
UDATA   23-14     23-16     24-11*    24-15     24-20#
UNIT0    4-46#
UNIT1    4-47#
UNIT2    4-48#
UNIT3    4-49#
UNSSUC   5-42#
UREAD    3-14#    23-15     35-19
UTOTST   5-13#
UWRITE   3-15#    24-16     34-44
WAITSI   3-13#
WBUFLN   2-119#    2-120
WCHECK   7-37#
WCHKAL   7-39#
```

CROSS REFERENCE TABLE (CREF V04.00)

```
WCONT     2-90#
WONLY     7-30#
WREAL     2-94#
WRITE    32-6      32-21     34-9#
WRONG     4-65#
WSTOP     2-89#
XBNCYL    4-41#
XFERRT    4-5#
XMTERR    5-25#    19-20
XOR      25-10#    30-9      30-20     37-44     37-48
XREAD     3-5#
XWRITE    3-6#
```

CROSS REFERENCE TABLE (CREF V04.00)

UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42
TABLE OF CONTENTS

```
                                .TITLE  UDAT2   DISK RESIDENT
                        ;
                        ;
                        ; COPYRIGHT (C)  1981
                        ; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.
                        ;
                        ;
                        ; THIS  SOFTWARE  IS  FURNISHED  UNDER A LICENSE FOR USE ONLY ON A
                        ; SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION
                        ; OF  THE  ABOVE COPYRIGHT  NOTICE.  THIS SOFTWARE, OR  ANY  OTHER
                        ; COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE  MADE AVAILABLE
                        ; TO  ANY  OTHER PERSON  EXCEPT  FOR USE ON SUCH SYSTEM AND TO ONE
                        ; WHO AGREES TO THESE LICENSE TERMS.  TITLE  TO  AND  OWNERSHIP OF
                        ; THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DIGITAL.
                        ;
                        ; THE  INFORMATION  IN THIS DOCUMENT IS SUBJECT  TO CHANGE WITHOUT
                        ; NOTICE AND SHOULD NOT BE CONSTRUED AS A  COMMITMENT  BY  DIGITAL
                        ; EQUIPMENT CORPORATION.
                        ;
                        ; DIGITAL ASSUMES NO  RESPONSIBILITY FOR THE USE OR RELIABILITY OF
                        ; ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;THIS PROGRAM SHALL BE ASSEMBLED WITH THE PROGRAM DMACRO
                        ;USING A COMMAND LINE SIMILAR TO:
                        ;
                        ;  UDAT2.BIN,UDAT2/C=[1,2]DMACRO,UDAT2
                        ;
                        ;       .ENABL  ABS
000000                          DMCODE  UDADM2,0,714,3,0,1000           ;41000
                        .SBTTL  UDA DM PROGRAM PARAMETERS
                        ;
                        ;       .LIST MEB
003720                          ERRN=2000.                              ;START ERROR NUMBERS AT 2000.
                        ;
                        ;       EQUATES
                        ;
                        ;       HIGHEST USABLE LOCATION OF UDA MEMORY + 1
                        ;
007774                  HIMEM   =       7774                    ; HIGH MEMORY+1
007774                  OVSTRT  =       7774                    ; OVERLAY ADDRESS LOCATION
                        ;
                        ;
                        ;       OFFSETS FOR FORMAT TRACK TABLE
                        ;
000000                  FT.BUF  =       0.                      ;BUFFER POINTER OFFSET
000001                  FT.HI   =       1.                      ;HI ORDER HEADER OFFSET
000002                  FT.LOW  =       2.                      ;LOW ORDER HEADER OFFSET
```

```
                                ;
                                ;
                                ;       OFFSETS FOR FORMAT TRACK BUFFER
                                ;
000000                          FB.DAT  =       0.                      ;FIRST DATA WORD OFFSET
000400                          FB.EDC  =       256.                    ;EDC WORD OFFSET
                                ;
                                ;
                                ;       OFFSETS FOR READ/WRITE I/O CHAIN TABLES
                                ;
000000                          RW.STAT =       0.                      ;STATUS AND NEXT BUFFER POINTER OFFSET
000001                          RW.BUF  =       1.                      ;POINTER TO DATA BUFFER
000002                          RW.LOW  =       2.                      ;HI ORDER EXPECTED HEADER
000003                          RW.HI   =       3.                      ;LOW ORDER EXPECTED HEADER
000004                          RW.CMD  =       4.                      ;SDI COMMAND AND HEAD ADDRESS
000005                          RW.SDI  =       5.                      ;DUMMY SDI CONTROL BLOCK POINTER
000006                          RW.ANG  =       6.                      ;THETA FROM INDEX
                                ;
                                ;       CONSTANTS FOR READ AND WRITE XFC'S
                                ;
140000                          WSTOP   =       140000                  ; LAST ENTRY IN CHAIN FOR WRITE
040000                          WCONT   =       40000                   ; WRITE CONTINUE
100000                          RSTOP   =       100000                  ; LAST ENTRY IN CHAIN FOR READ
000000                          RCONT   =       0                       ; READ CONTINUE
100000                          FSTOP   =       100000                  ; LAST ENTRY IN CHAIN FOR FORMAT
122400                          WREAL   =       122400                  ; WRITE REAL TIME ECOMMAND
013400                          RREAL   =       13400                   ; READ REAL TIME COMMAND
010000                          ECCFLG  =       10000                   ; ECC ERROR IN BUFFER BIT
100000                          EOC     =       100000                  ; END OF CHAIN
040000                          BUFFLG  =       40000                   ; BUFFER FULL OR EMPTY FLAG
                                ;
                                ;
                                ;       HEADER CODES
                                ;
000000                          HD.LBN  =       000000                  ;GOOD LBN
060000                          HD.RBN  =       060000                  ;GOOD RBN, PERHAPS UNUSED
030000                          HD.REV  =       030000                  ;REVECTORED LBN
110000                          HD.BAD  =       110000                  ;BAD BLOCK
050000                          HD.PRV  =       050000                  ;PRIMARY REVECTORED BLOCK
120000                          HD.XBN  =       120000                  ;XBN BLOCK
140000                          HD.DBN  =       140000                  ;DBN BLOCK

                                ;       OFFSETS FOR DATA BUFFERS

000000                          BF.DAT  =       0.                      ;DATA
000400                          BF.EDC  =       256.                    ;ERROR DETECTION CODE
000401                          BF.ECC  =       257.                    ;LAST 17 ECC RESIDUES

                                ;       BUFFER AND READ/WRITE CHAIN LINK SIZES

000401                          WBUFLN  =       257.                    ; WRITE BUFFER SIZE
000415                          RBUFLN  =       WBUFLN+12.              ; READ BUFFER SIZE
000007                          LINKLN  =       7.                      ; LINK SIZE
```

```
                                ;       XFC DEFINITION EQUATES
                                ;
000000                          BREAK   =       0.      ;BREAKPOINT XFC CODE
000001                          FORMAT  =       1.      ;FORMAT TRACK XFC CODE
000002                          XREAD   =       2.      ;READ N SECTORS XFC CODE
000003                          XWRITE  =       3.      ;WRITE N SECTORS XFC CODE
000004                          SEND    =       4.      ;SEND SDI COMMAND XFC CODE
000005                          RCV     =       5.      ;RECEIVE SDI MESSAGE XFC CODE
000006                          COMPARE =       6.      ;COMPARE DATA PATTERN TO BUFFER
000007                          STATUS  =       7.      ;RETURN DRIVE STATUS XFC CODE
000010                          ECHO    =       8.      ;ECHO DATA TO DRIVE XFC CODE
000011                          DINIT   =       9.      ;DRIVE INITIALIZE XFC CODE
000012                          WAITSI  =       10.     ;WAIT FOR SECTOR OR INDEX PULSE
000013                          UREAD   =       11.     ;READ UNIBUS MEMORY XFC CODE
000014                          UWRITE  =       12.     ;WRITE UNIBUS MEMORY XFC CODE
000015                          ECC     =       13.     ;DO ECC ON BUFFER XFC CODE
000016                          MRD     =       14.     ;SEND BUFFER TO MAINTENANCE READ COMMAND
000017                          MWR     =       15.     ;GET BUFFER FROM MAINTENANCE WRITE COMMAND
000020                          CVT     =       16.     ;CONVERT TO PHYSICAL ADDRESS XFC CODE
000021                          EXIT    =       17.     ;TERMINATE DM PROGRAM XFC CODE
                                ;
                                ;       GET STATUS OFFSETS
                                ;
000000                          ST.UNT  =       0.      ;UNIT NUMBER
000000                          ST.MSK  =       0.      ;SUBUNIT MASK
000001                          ST.STA  =       1.      ;STATUS BYTE
000001                          ST.MOD  =       1.      ;MODE BYTE
000002                          ST.ERR  =       2.      ;ERROR BYTE
000002                          ST.CON  =       2.      ;CONTROLLER BYTE
000002                          ST.C    =       2.      ;C BITS
000003                          ST.RTY  =       3.      ;RETRY COUNT/FAILURE CODE
                                ;
                                ;       STATUS BIT DEFINITIONS
                                ;
000200                          ST.OA   =       200     ; ONLINE TO ANOTHER (SET IF DRIVE UNAVAILABLE)
000100                          ST.RR   =       100     ; READJUSTMENT BIT (SET IF RECALIBRATION REQUIRED)
000040                          ST.DR   =       40      ; DIAGNOSTIC REQUEST (SET IF DIAGNOSTIC REQUESTED)
000020                          ST.SR   =       20      ; SPINDLE READY (SET IF SPINDLE READY)
000002                          ST.PS   =       2       ; PORT SWITCH (SET IF PORT SWITCH IN)
000001                          ST.RU   =       1       ; RUN/STOP SWITCH (SET IF RUN/STOP SWITCH IN)
000200                          ST.FE   =       200     ; FATAL ERROR (SET IF FATAL ERROR OCCURRED)
000100                          ST.RE   =       100     ; RETRIABLE ERROR (SET IF RETRIABLE ERROR OCCURRED)
000040                          ST.PE   =       40      ; PROTOCOL ERROR (SET IF PROTOCOL ERROR OCCURRED)
000020                          ST.DF   =       20      ; INITIALIZATION FAILURE (SET IF INIT FAILED)
000010                          ST.WE   =       10      ; WRITE ENABLE (SET IF WRT ATTEMPTED ON PROT DISK)
002000                          ST.FO   =       2000    ; FORMATTING (SET IF FORMATTING ENABLED)
001000                          ST.DB   =       1000    ; DIAGNOSTIC CYLS (SET IF DIAG CYL ACCESS ENABLED)
000400                          ST.S7   =       400     ; SECTOR SIZE (SET FOR 576 BYTE SECTORS)
```

```
                                ;       GET COMMON CHARACTERISTICS OFFSETS
                                ;
000000                          SHRTTO  =       0.              ;SHORT TIMEOUT <3:0>
000000                          SDIVER  =       0.              ;SDI VERSION <7:4>
000000                          XFERRT  =       0.              ;TRANSFER RATE <15:0>
000001                          LONGTO  =       1.              ;LONG TIMEOUT <3:0>
000001                          RETS    =       1.              ;RETRIES <7:4>
000001                          RCTCPS  =       1.              ;F/RCT COPIES <11:8>
000001                          SS      =       1.              ;SECTOR SIZE <15:15>
000002                          ERLEV   =       2.              ;ERROR RETRY LEVELS <7:0>
000002                          ECCRSH  =       2.              ;ECC THRESHOLD <15:8>
000003                          MICREV  =       3.              ;MICROCODE REVISION NUMBER <7:0>
000003                          HRDREV  =       3.              ;HARDWARE REVISION NUMBER <15:8>
000004                          DRVID   =       4.              ;UNIQUE DRIVE ID <47:0>
000007                          DRTYPE  =       7.              ;DRIVE TYPE IDENTIFIER <7:0>
000007                          REVS    =       7.              ;REVS/SECOND <15:8>
                                ;
                                ;       GET SUBUNIT CHARACTERISTICS OFFSETS
                                ;
                                ;THESE OFFSETS ARE CURRENTLY GIVEN AS FOLLOWING THE COMMON CHARACTERISTICS
                                ;
000013                          SUB     =       11.             ;OFFSET TO PUT SUBUNIT AFTER COMMON;
000000                          LBNCYL  =       0.              ;NUMBER OF CYLINDERS IN LBN AREA <31:0>
000001                          HICYL   =       1.              ;HI ORDER CYLINDER BITS <15:12>
000002                          GRPCYL  =       2.              ;GROUPS PER CYLINDER <7:0>
000002                          HILBN   =       2.              ;HI STARTING LBN <11:8>
000002                          HIXBN   =       2.              ;HI STARTING XBN <15:12>
000003                          TRKGRP  =       3.              ;TRACKS PER GROUP <7:0>
000003                          HIRBN   =       3.              ;HI STARTING RBN <11:8>
000003                          HIDBN   =       3.              ;HI STARTING DBN <15:12>
000004                          RBNTRK  =       4.              ;RBNS PER TRACK <6:0>
000004                          RM      =       4.              ;REMOVABLE MEDIA <7:7> 1=REMOVEABLE
000005                          DATPRE  =       5.              ;DATA PREAMBLE SIZE IN WORDS <7:0>
000005                          HDRPRE  =       5.              ;HEADER PREAMBLE SIZE IN WORDS <15:8>
000006                          MEDTYP  =       6.              ;MEDIA TYPE <31:0>
000010                          FCTSIZ  =       8.              ;FCT COPY SIZE <15:0>
000011                          LBNTRK  =       9.              ;LBNS PER TRACK <7:0>
000011                          GRPOFF  =       9.              ;GROUP OFFSET (SECTORS) <15:8>
000012                          LBNHST  =       10.             ;LBNS IN HOST AREA <31:0>
000014                          RCTCSZ  =       12.             ;RCT COPY SIZE <15:0>
000021                          XBNCYL  =       17.             ;CYLS IN XBN AREA <15:0>
000022                          DBNCYL  =       18.             ;CYLS IN DBN AREA <15:8>
                                ;
                                ;       UNIT CODES
                                ;
000001                          UNIT0   =       1.              ;UNIT ZERO CODE
000002                          UNIT1   =       2.              ;UNIT ONE CODE
000004                          UNIT2   =       4.              ;UNIT TWO CODE
000010                          UNIT3   =       8.              ;UNIT THREE CODE
                                ;
                                ;       BIT MASK DEFINITIONS
                                ;
                                ;
177400                          HIBYTE  =       177400          ;HIGH BYTE MASK
000377                          LOBYTE  =       000377          ;LOW BYTE MASK
007777                          HBHINB  =       7777            ;HI BYTE, HI NIBBLE MASK
170377                          HBLONB  =       170377          ;HI BYTE, LO NIBBLE MASK
```

```
                177417          LBHINB  =       177417          ;LO BYTE, HI NIBBLE MASK
                177760          LBLONB  =       177760          ;LO BYTE, LO NIBBLE MASK
                                ;
                000001          TIMEOUT =       1.              ;DRIVE TIMEOUT CODE
                000002          HEADER  =       2.              ;HEADER COMPARE FAILURE CODE
                000004          REVECT  =       4.              ;REVECTOR NEEDED CODE
                                ;
                000002          WRONG   =       2.              ;FIRST WORD NOT START FRAME CODE
                000004          FRAME   =       4.              ;FRAMING ERROR CODE
                000010          CHECK   =       8.              ;CHECKSUM ERROR CODE
                                ;
                000001          TOOBIG  =       1.              ;NUMBER OF WORDS EXCEEDS 7064
                000002          LOW     =       2.              ;DM BUFFER ADDRESS IS LESS THAN 714
                                ;
                                ;       BIT MASK
                                ;
                000001          LARGE   =       1.              ;BLOCK NUMBER TOO LARGE
                000002          OVERFL  =       2.              ;SECTOR NUMBER LARGER THAN 16 BITS
```

```
                                ;MAINTANENCE READ/WRITE REQUEST NUMBERS
                                ;
060000          T1MSIZ  =       0.+DU.SPC               ;GET FREE MEMORY PARAMETERS
060001          T2DLL   =       1.+DU.SPC               ;DOWNLINE LOAD DRIVE DIAGNOSTIC
060002          T2CMD   =       2.+DU.SPC               ;MANUAL INTERVENTION TEST 2 PROTOCOL
060003          T4MPRM  =       3.+DU.SPC               ;GET MASTER PARAMETERS FROM SW QUESTIONS
060004          T4UPRM  =       4.+DU.SPC               ;GET UNIT PARAMETERS FROM HW QUESTIONS
060005          T4BB1   =       5.+DU.SPC               ;GET BAD BLOCKS (1 THRU 14)
060006          T4BB2   =       6.+DU.SPC               ;GET REST OF BAD BLOCKS (15 AND 16)
060007          T4SOFT  =       7.+DU.SPC               ;ADD TO SOFT ERROR AND ECC COUNT
060010          T4SEEK  =       8.+DU.SPC               ;ADD 1 TO SEEK COUNT
060011          T4MXFR  =       9.+DU.SPC               ;ADD TO MEGABITS READ AND WRITTEN
060012          UTOTST  =       10.+DU.SPC              ;GET UNITS TO TEST
060013          ERRMES  =       11.+DU.SPC              ;PRINT ERROR MESSAGE
060014          ERRMC   =       12.+DU.SPC              ;TEST 4 ERROR REPORTING
060015          MESSAG  =       13.+DU.SPC              ;INFORMATION MESSAGE
060016          DONE    =       14.+DU.SPC              ;MARK DM PROGRAM AS NO LONGER RUNNING
                                ;
                                ;       OTHER BIT DEFINITIIONS
                                ;
000001          RCVRDY  =       1                       ; RECIEVER READY 1 = READY
000002          ATTN    =       2                       ; ATTENTION BIT FOR RETURN DRIVE SIGNALS XFC
000004          RCVERR  =       4                       ; RECIEVER ERROR
000100          AVAIL   =       100                     ; AVAILABLE 1 = AVAILABLE
000400          XMTERR  =       400                     ; TRANSMIT ERROR
100000          RWRDY   =       100000                  ; IF SET, UDA IS ABLE TO READ AND/OR WRITE TO DRIVE
                                ;
                                ;       SDI COMMANDS AND RESPONSES
                                ;
000204          DISCON  =       204                     ; DISCONNECT DRIVE
000006          ERECOV  =       6                       ; ERROR RECOVERY
000201          CHGMOD  =       201                     ; CHANGE MODE
000213          DRVONL  =       213                     ; DRIVE ONLINE
000014          DRVRUN  =       14                      ; DRIVE RUN
000005          DRVCLR  =       5                       ; DRIVE CLEAR OPCODE
000207          GETCHR  =       207                     ; GET CHARACTERISTICS
000210          GETSUB  =       210                     ; GET SUBUNIT CHARACTERISTICS
000011          GETSTA  =       11                      ; GET STATUS
000216          IRECLB  =       216                     ; RECALIBRATE
000012          INSEEK  =       12                      ; INITIATE SEEK
000176          COMPLT  =       176                     ; SUCCESSFUL COMPLETION
000175          UNSSUC  =       175                     ; UNSUCCESSFUL COMPLETION
000170          CHRRES  =       170                     ; GET CHARACTERISTICS RESPONSE
000167          SBCRES  =       167                     ; GET SUBUNIT CHARACTERISTICS RESPONSE
000366          STSRES  =       366                     ; GET STATUS RESPONSE
000350          ECHOC   =       350                     ; DIAGNOSTIC ECHO COMMAND AND RESPONSE
                                ;
                                ;       ERROR CODES
                                ;
000000          FTLSYS  =         0                     ; SYSTEM FATAL ERROR
040000          FTLDEV  =     40000                     ; DEVICE FATAL
100000          ERHARD  =    100000                     ; HARD ERROR
140000          ERSOFT  =    140000                     ; SOFT ERROR
```

```
                                ;       DUMMY SDI CONTROL BLOCK OFFSETS
                                ;
000001                          D.LIMT  =       1                       ; DUMMY SDI SEARCH LIMIT
000002                          D.SCHR  =       2                       ; DUMMY POINTER TO SUBUNIT CHAR-5
                                ;
                                ;       DUP MESSAGE TYPES
010000                          DU.QUE = 10000
020000                          DU.DFL = 20000
030000                          DU.INF = 30000
040000                          DU.TER = 40000
050000                          DU.FTL = 50000
060000                          DU.SPC = 60000
                                .SBTTL  MACRO DEFINITIONS
                                ;
                                ;       MESSAGE CONTROL TABLE MACRO
                                ;
                                        .MACRO MSG CMDBUF,CMDSZ,RPLBUF,RPLSZ,SUCCOM
                                        .WORD CMDBUF                    ;ADDRESS OF COMMAND
                                        .WORD CMDSZ                     ;SIZE OF COMMAND IN BYTES
                                        .WORD RPLBUF                    ;ADDRESS OF REPLY
                                        .WORD RPLSZ                     ;SIZE OF REPLY IN WORDS
                                        .IF NB NUMBER
                                        .WORD SUCCOM                    ; SUCCESSFUL COMPLETION CODE
                                        .ENDC
                                        .ENDM
```

MACRO DEFINITIONS

```
      1                                .MACRO  BCS     LAB..
      2                                                        BCC     .+2
      3                                                        BR      LAB..
      4                                .ENDM
```

```
UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42 PAGE 8
MACRO DEFINITIONS

                        ;       PUSH REGISTER MACRO
                        ;
                                .MACRO PUSH R9
                                .IRP X,<R9>
                                MOV X,-(SP)
                                .ENDR
                                .ENDM
                        ;
                        ;       POP REGISTER MACRO
                        ;
                                .MACRO POP R9
                                .IRP X,<R9>
                                MOV (SP)+,X
                                .ENDR
                                .ENDM
```

```
                        ;ERROR MACROS
                        ;THESE MACROS ARE CALLED TO REPORT ERRORS TO THE HOST PROGRAM.
                        ;THE MACRO NAMES ARE : ERRSF, ERRDF, ERRHRD, ERRSFT. EACH RESULTS IN THE HOST
                        ;BEING REQUESTED TO REPORT THE ERROR.
                        ;ARGUMENTS:     1 (MS$) MESSAGE POINTER
                        ;               2 (P1$) PARAMETER #1
                        ;               3 (P2$) PARAMETER #2
                        ;               4 (P3$) PARAMETER #3
                        ;               5 (P4$) PARAMETER #4
                        ;               6 (P5$) PARAMETER #5
                        ;               7 (P6$) PARAMETER #6
                        ;               8 (P7$) PARAMETER #7
                        ;               9 (P8$) PARAMETER #8
                        ;
                        ;THE MESSAGE POINTER MUST POINT TO AN ADDRESS IN THE OVERLAY 'MS' IMMEDIATELY
                        ;FOLLOWING THE MAIN CODE. ANY ADDRESS MODE MAY BE USED (E.G. #MS1, @R2).
                        ;THE ADDRESS MUST CONTAIN AN ASCII FORMAT STRING TO DETERMINE THE MESSAGE
                        ;TO PRINT.
                        ;THE PARAMETER ARGUMENTS ARE OPTIONAL. THEY SHOULD BE SUPPLIED ONLY WHEN
                        ;THERE IS DATA TO BE PASSED TO THE HOST THAT WILL BE USED IN PRINTING THE
                        ;MESSAGE. THESE PARAMETER ARGUMENTS ARE THE ADDRESS OF DATA TO BE PASSED
                        ;USING ANY ADDRESSING MODE DESIRED.
                        ;ALL REGISTERS ARE RETURNED UNCHANGED. IT SHOULD BE NOTED THAT ARGUMENTS
                        ;CONTAINING SOMETHING OTHER THAN A REGISTER NAME (E.G. #100 OR MEMADR)
                        ;ASSEMBLE TO INSTRUCTIONS THAT SAVE AND RESTORE A REGISTER ON THE STACK.

                                .MACRO ERRSF MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$
                                .NARG ARGS$
                                .RADIX 10
                                ERROR$ FTLSYS,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,\ERRN
                                .ENDM

                                .MACRO ERRDF MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$
                                .NARG ARGS$
                                .RADIX 10
                                ERROR$ FTLDEV,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,\ERRN
                                .ENDM

                                .MACRO ERRHRD MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$
                                .NLIST
                                .NARG ARGS$
                                .RADIX 10
                                ERROR$ ERHARD,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,\ERRN
                                .LIST
                                .ENDM

                                .MACRO ERRSFT MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$
                                .NARG ARGS$
                                .RADIX 10
                                ERROR$ ERSOFT,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,\ERRN
                                .ENDM
```

```
                        ;THE FOLLOWING MACRO ACTUALLY PROCESSES THE ERROR CALL TO THE HOST PROGRAM

                .MACRO ERROR$ ET$,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,ERRN$
                  .RADIX 8
                  PRMS=ARGS$-1
                  .IF LT,<PRMS>
                    .ERROR;NOT ENOUGH ARGUMENTS IN ERROR CALL
                  .ENDC
                  REGS$=-1
                  .IIF GE,<PRMS-8.>,PARGS. P8$
                  .IIF GE,<PRMS-7.>,PARGS. P7$
                  .IIF GE,<PRMS-6.>,PARGS. P6$
                  .IIF GE,<PRMS-5.>,PARGS. P5$
                  .IIF GE,<PRMS-4.>,PARGS. P4$
                  .IIF GE,<PRMS-3.>,PARGS. P3$
                  .IIF GE,<PRMS-2.>,PARGS. P2$
                  .IIF GE,<PRMS-1.>,PARGS. P1$
                  .IF GE REGS$
                    RSTR$ \REGS$
                  .ENDC
                  .RADIX 10
                  .LIST
                                                                CALL RERROR     ;ERROR # ERRN$'.
                  .NLIST
                  .RADIX 8
                  .LIST
                                                                .WORD ET$+ERRN
                                                                .WORD <PRMS*10000>+MS$
                  .NLIST
                  ERRN=ERRN+1
                .ENDM

                .MACRO PARGS.,ADDR$
                  .NTYPE PTYPE$,ADDR$
                  .IF EQ,<PTYPE$&70>
                    .IIF EQ,<PTYPE$&7>-REGS$,RSTR$ \REGS$
                    .LIST
                                                                MOV ADDR$,-(SP)
                    .NLIST
                  .IFF
                    .IF EQ,<PTYPE$&7>-1         ;PICK A REGISTER TO USE
                      REGU$=2                   ;SELECT R2 IF R1 IS USED IN PARAMETER FETCH
                    .IFF
                      REGU$=1                   ;OTHERWISE USE R1
                    .ENDC
                    .IF NE,<REGU$-REGS$>        ;IF REGISTER NOT ALREADY SAVED
                      .IF GE,REGS$
                        RSTR$ \REGS$            ;RESTORE CURRENT SAVED REGISTER
                      .ENDC
                      SAVR$ \REGU$              ;THEN SAVE SELECTED REGISTER
                    .ENDC
                    GETP$ \REGS$,ADDR$
                  .ENDC
                .ENDM

                .MACRO  DEVFTL  NUM,ARGS
                ERROR   FTLDEV,NUM,<ARGS>
```

```
                                                                    MOV     #ERRMC,OUT.RQ
                        .ENDM
                        .MACRO  ERROR   TYPE,NUM,ARGS
                        .RADIX  10
                        NUMPTR  =       5
                        MOVMSG  #ER'NUM,4
                        .IF     NB,<ARGS>
                        .IRP    X,<ARGS>
                        MOVMSG  X,\NUMPTR
                        NUMPTR  =       NUMPTR + 1
                        .ENDR
                        .ENDC
                                                                    MOV     #NUM!TYPE,R2
                                                                    MOV     R2,OUT.02
                                                                    MOV     #.,OUT.01
                        .RADIX
                        .ENDM
```

MACRO DEFINITIONS

```
                                .MACRO SAVR$ REGN
                                  .LIST
                                                                MOV R'REGN,SAVREG
                                  .NLIST
                                  REGS$=REGN
                                .ENDM

                                .MACRO RSTR$ REGN
                                  .LIST
                                                                MOV SAVREG,R'REGN
                                  .NLIST
                                  REGS$=-1
                                .ENDM

                                .MACRO GETP$ REGN,ADDR$
                                  .LIST
                                                                MOV ADDR$,R'REGN
                                                                MOV R'REGN,-(SP)
                                  .NLIST
                                .ENDM
```

MACRO DEFINITIONS

```
                        .MACRO  MSSGE   NUM,ARGS
                        .RADIX  10
                        NUMPTR  =       3

                                                        MOV     LUNIT,OUT.01
                        MOVMSG  #'NUM,2
                        .IF     NB,<ARGS>
                        .IRP    X,<ARGS>
                        MOVMSG  X,\NUMPTR
                        NUMPTR  =       NUMPTR + 1
                        .ENDR
                        .ENDC

                                                        PUSH    <R0,R1>
                                                        MOV     #MESSAG,R0
                                                        CALL    HOSTRQ
                                                        POP     <R1,R0>

                        .RADIX
                        .ENDM

                        .MACRO  MOVMSG  ARG,INDX

                        .IF     LT,INDX-10

                                                        MOV     ARG,OUT.0'INDX
                        .IFF

                                                        MOV     ARG,OUT.'INDX
                        .ENDC
                        .ENDM
```

MACRO DEFINITIONS

```
      1                                 ;ASSUME MACRO
      2                                         .MACRO  ASSUME  P1,P2
      3                                         .IF     NE,P1-P2
      4                                         .ERROR  ;       THE ASSUMPTION THAT THE ABOVE VALUES ARE = IS FALSE
      5                                         .ENDC
      6                                         .ENDM
```

```
                        ;RETURN DRIVE STATUS MACRO WITH ERROR REPORTING

                                .MACRO  DSTAT,LAB$,E1,E2
                                .NLIST
                                .NLIST  MEB
                                .LIST   ME
                                .LIST
                                CALL    RDSTAT                  ; GET DRIVE STATUS
                                BIT     #10000,R1               ; SEE IF ANY ERRORS
                                BEQ     2$                      ; IF NO ERROR, BRANCH
                                BIT     #4000,R1                ; SEE IF XMIT ERROR
                                BEQ     1$                      ; IF SO, BRANCH
                                ERRHRD  E1                      ; REPORT INVALID STATUS ERROR
                                BR      LAB$                    ; BRANCH TO DONE
                        1$:     ERRHRD  E2                      ; REPORT XMIT ERROR
                                BR      LAB$                    ; BRANCH TO DONE
                        2$:
                                .NLIST
                                .NLIST  ME
                                .LIST   MEB
                                .LIST
                                .ENDM
```

```
1                                       ;
2                                       ;       XOR THE CONTENTS OF TWO REGISTERS
3                                       ;
4                                               .MACRO  RXOR    REG1,REG2
5                                               MOV     REG2,-(SP)              ; SAVE REGISTER REG2
6                                               BIC     REG1,REG2               ; CLEAR COORESPONDING BITS IN REG2
7                                               BIC     (SP)+,REG1              ; CLEAR COORESPONDING BITS IN REG1
8                                               BIS     REG1,REG2               ; OR WHAT'S LEFT
9                                               .ENDM
```

```
                        ;
                        ;       CONVERT .BLKW CALLS TO ACTUAL WORDS GENERATED
                        ;
                        .MACRO  .BLKW   COUNT
                        .NLIST
                        .REPT   COUNT
                        .WORD   0
                        .ENDR
                        .LIST
                        .ENDM
```

```
                        ;       SDI INTERCHANGE WITH DRIVE WITH ERROR REPORTING
                        
                                .MACRO  TALKX   ERRLAB,E1,E2
                                .NLIST
                                .NLIST  MEB
                                .LIST   ME
                                .LIST
                                CALL    TALKER                  ; INITIATE SDI INTERCHANGE
                                TST     R3                      ; SEE IF ERROR OCCURRED
                                BEQ     12$                     ; IF NOT, BRANCH
                                BPL     11$                     ; IF SO, BRANCH
                                ERRHRD  E1;SEND COMMAND ERROR
                                BR      ERRLAB
                        11$:    ERRHRD  E2;RECEIVE COMMAND ERROR
                                BR      ERRLAB
                        12$:
                                .NLIST
                                .NLIST  ME
                                .LIST   MEB
                                .LIST
                                .ENDM
```

```
UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42 PAGE 18
START OF TEST CODE

                                .SBTTL  START OF TEST CODE
                                ;THE FOLLOWING IS FOR DEBUG PURPOSES ONLY. CAN BE NOP OR BREAKPOINT.
                                
000714  114007                          CLR R0                          ;CHANGE TO BREAKPOINT FOR DEBUG
                                
                                ;INITIALIZE STACK
                                
000715  104206  001646                  MOV #STACK,SP                   ;SET UP STACK POINTER
000717  001647                          BR      START                   ; BRANCH OVER SUPPORT CODE
```

```
                                                .SBTTL  PROGRAM VARIABLES
                                        ;PROGRAM VARIABLES
                                        ;UNIT NUMBER STORAGE FOR DISK DRIVES TO TEST
000720  177777  177777  177777  UNITS:  .WORD   -1,-1,-1,-1             ;LOGICAL UNIT NUMBER IF POSITIVE
000723  177777
000724  177777  177777  177777          .WORD   -1,-1,-1,-1             ;DISK DRIVE NOT TO BE TESTED IF NEGATIVE
000727  177777
000730  177777  177777  177777          .WORD   -1,-1,-1,-1
000733  177777
000734  177777  177777  177777          .WORD   -1,-1,-1,-1
000737  177777

000740  000000          UNITNB: .WORD   0                               ;NUMBER OF UNIT CURRENTLY UNDER TEST
                                                                        ;POINTER INTO TABLE ABOVE
000741  000000          SDI:    .WORD   0                               ;SDI INTERCONNECT CODE FOR XFC CALLS
000742  000000          SAVSTA: .WORD   0                               ;SAVE STATUS
000743  000000          SAVRID: .WORD   0                               ;SAVE REGION ID
                        ;DRIVE RESPONSE BUFFERS
        000200          STSIZE  =       200
000744                  ST:     .BLKW   200
                        ;
001144  000000          DIAG.1: .WORD   0                               ; = 0, 1ST DIAGNOSE COMMAND
                                                                        ; NOT = 0, DRIVE TESTED BEFORE
001145  000000          NAM.1:  .WORD   0                               ;HOLD PROGRAM NAME
001146  000000          NAM.2:  .WORD   0
001147  000000          NAM.3:  .WORD   0
001150  000000                  .WORD   0
                        ;
001151  000000          SENDHR: .WORD   0                               ;IF 0, DON'T SEND HOSTRQ IN TALKER
                                                                        ;IF NOT 0, SEND HOSTRQ -> DOING DIAGNOSE COMMAND
                        ;
001152  177777          LUNIT:  .WORD -1                                ;LOGICAL UNIT NUMBER (-1 FOR NOT AVAILABLE)

001153  000000          SAVREG: .WORD   0                               ;STORAGE FOR REGISTER AT CALL TIME
001154  000012          SDISTO: .WORD   10.                             ; SDI SHORT TIMEOUT
001155  000024          SDILTO: .WORD   20.                             ; SDI LONG TIMEOUT

001156  004212          SER18E: .WORD   SER18A                          ; FOR MULTIPLE SUBUNIT ERROR REPORTING
001157  004206                  .WORD   SER18B
001160  004202                  .WORD   SER18C
001161  004176                  .WORD   SER18D

        001750          MAXSND  =       1000.                           ; MAXIMUM # OF SENDS

        000017          WRM.OP  =       17                              ; WRITE MEMORY COMMAND
        000215          RDM.OP  =       215                             ; READ MEMORY COMMAND
        000003          DIA.OP  =       3                               ; DIAGNOSE COMMAND
        000207          GETCHR  =       207                             ; GET COMMON CHARACTERISTICS COMMAND
        000162          RDM.EN  =       162                             ; READ MEMORY RESPONSE
        000374          DIA.EN  =       374                             ; DIAGNOSE RESPONSE
        000170          GCC.EN  =       170                             ; GET COMMON CHARACTERISTICS
```

```
                                ;
                                ;       DIAGNOSE ET AND DA
                                ;
100000                          ERRTYP  =       100000          ;ERROR TYPE ET
040000                          DATAVL  =        40000          ;DATA AVAILABLE
                                ;
                                ;       REGION ID'S
                                ;
177774                          FFFC    =       FFFD - 1
177775                          FFFD    =       177775
177776                          FFFE    =       FFFD + 1
```

```
                                                .SBTTL  SDI COMMANDS USED FOR TEST 2
001162      377     350         ECHOD:  .BYTE   377,ECHOC                ;ECHO DATA TO SEND TO DRIVE
001163      000     350                 .BYTE   000,ECHOC
001164      252     350                 .BYTE   252,ECHOC
001165      360     350                 .BYTE   360,ECHOC
001166      017     350                 .BYTE   017,ECHOC
001167   000000                         .WORD   0                       ;END MARKER

                                                .SBTTL  COMMAND BUFFERS FOR SEND XFC
                                        ;COMMAND BUFFERS FOR SEND XFC

001170                                  CR.GST: MSG     GETST,1,ST,7            ; GET STATUS SDI INFORMATION
001170   001175                                 .WORD GETST                     ;ADDRESS OF COMMAND
001171   000001                                 .WORD 1                 ;SIZE OF COMMAND IN BYTES
001172   000744                                 .WORD ST                        ;ADDRESS OF REPLY
001173   000007                                 .WORD 7                 ;SIZE OF REPLY IN WORDS
001174   000000                                 .WORD                   ; SUCCESSFUL COMPLETION CODE
001175      000     011         GETST:  .BYTE   0,GETSTA
                                        ;
001176                                  CR.DRC: MSG     DRC,2,ST,7              ; DRIVE CLEAR SDI INFORMATION
001176   001203                                 .WORD DRC                       ;ADDRESS OF COMMAND
001177   000002                                 .WORD 2                 ;SIZE OF COMMAND IN BYTES
001200   000744                                 .WORD ST                        ;ADDRESS OF REPLY
001201   000007                                 .WORD 7                 ;SIZE OF REPLY IN WORDS
001202   000000                                 .WORD                   ; SUCCESSFUL COMPLETION CODE
001203      000     005         DRC:    .BYTE   0,DRVCLR
001204   177777                 DRCBYT: .WORD   177777
                                        ;
001205                                  CR.WRM: MSG     WRM,7,ST,7              ; MEMORY WRITE COMMAND INFO
001205   001212                                 .WORD WRM                       ;ADDRESS OF COMMAND
001206   000007                                 .WORD 7                 ;SIZE OF COMMAND IN BYTES
001207   000744                                 .WORD ST                        ;ADDRESS OF REPLY
001210   000007                                 .WORD 7                 ;SIZE OF REPLY IN WORDS
001211   000000                                 .WORD                   ; SUCCESSFUL COMPLETION CODE
001212      000      017        WRM:    .BYTE   0,WRM.OP
001213   000000  000000  000000                .WORD   0,0,0
001216   000000  000000  000000                .WORD   0,0,0
001221                          BUF1:   .BLKW   200
                                        ;
001421                                  CR.RDM: MSG     RDM,6,ST,STSIZE         ; MEMORY READ COMMAND INFO
001421   001426                                 .WORD RDM                       ;ADDRESS OF COMMAND
001422   000006                                 .WORD 6                 ;SIZE OF COMMAND IN BYTES
001423   000744                                 .WORD ST                        ;ADDRESS OF REPLY
001424   000200                                 .WORD STSIZE                    ;SIZE OF REPLY IN WORDS
001425   000000                                 .WORD                   ; SUCCESSFUL COMPLETION CODE
001426      000      215        RDM:    .BYTE   0,RDM.OP
001427   000000  000000  000000                .WORD   0,0,0,0
001432   000000
001433   000000  000000  000000                .WORD   0,0,0,0
001436   000000
                                        ;
001437                                  CR.GCR: MSG     GCR,1,ST,12.            ; GET CHARACTERISTICS
001437   001444                                 .WORD GCR                       ;ADDRESS OF COMMAND
001440   000001                                 .WORD 1                 ;SIZE OF COMMAND IN BYTES
001441   000744                                 .WORD ST                        ;ADDRESS OF REPLY
001442   000014                                 .WORD 12.                       ;SIZE OF REPLY IN WORDS
001443   000000                                 .WORD                   ; SUCCESSFUL COMPLETION CODE
```

```
     31 001444     000     207          GCR:    .BYTE   0,GETCHR                ; GET CHARACTERISTICS
     32                                 ;
     33 001445                          CR.ONL: MSG     ONL,2,ST,7              ; ONLINE COMMAND
        001445  001452                          .WORD ONL                       ;ADDRESS OF COMMAND
        001446  000002                          .WORD 2                 ;SIZE OF COMMAND IN BYTES
        001447  000744                          .WORD ST                        ;ADDRESS OF REPLY
        001450  000007                          .WORD 7                 ;SIZE OF REPLY IN WORDS
        001451  000000                          .WORD                   ; SUCCESSFUL COMPLETION CODE
     34 001452     000     213          ONL:    .BYTE   0,213
     35 001453  000077                          .WORD   77
     36                                 ;
     37 001454                          LONG:                                   ; ALL LONG TIMEOUT COMMANDS FOLLOW
     38                                 ;
     39 001454                          CR.DIA: MSG     DIA,3,ST,7              ; DIAGNOSE COMMAND INFORMATION
        001454  001461                          .WORD DIA                       ;ADDRESS OF COMMAND
        001455  000003                          .WORD 3                 ;SIZE OF COMMAND IN BYTES
        001456  000744                          .WORD ST                        ;ADDRESS OF REPLY
        001457  000007                          .WORD 7                 ;SIZE OF REPLY IN WORDS
        001460  000000                          .WORD                   ; SUCCESSFUL COMPLETION CODE
     40 001461     000     003          DIA:    .BYTE   0,DIA.OP
     41 001462  000000                          .WORD   0
     42 001463  000000                          .WORD   0
     43 001464                          CR.RUN: MSG     RUN,1,ST,7
        001464  001471                          .WORD RUN                       ;ADDRESS OF COMMAND
        001465  000001                          .WORD 1                 ;SIZE OF COMMAND IN BYTES
        001466  000744                          .WORD ST                        ;ADDRESS OF REPLY
        001467  000007                          .WORD 7                 ;SIZE OF REPLY IN WORDS
        001470  000000                          .WORD                   ; SUCCESSFUL COMPLETION CODE
     44 001471     000     014          RUN:    .BYTE   0,DRVRUN
     45 001472                          CR.INR: MSG     INR,1,ST,7
        001472  001477                          .WORD INR                       ;ADDRESS OF COMMAND
        001473  000001                          .WORD 1                 ;SIZE OF COMMAND IN 3YTES
        001474  000744                          .WORD ST                        ;ADDRESS OF REPLY
        001475  000007                          .WORD 7                 ;SIZE OF REPLY IN WORDS
        001476  000000                          .WORD                   ; SUCCESSFUL COMPLETION CODE
     46 001477     000     216          INR:    .BYTE   0,IRECLB
```

```
                                        .SBTTL  STORAGE AREA FOR MAINTENANCE WRITE AND READ BUFFERS
                                ;STORAGE AREA FOR MAINTENANCE WRITE AND READ BUFFERS
                                ;
                                ;OUT BUFFER - DATA TO SEND TO HOST
                                ;
001500  000000                  OUT.RQ: .WORD 0                 ;HOST REQUEST CODE
001501  000000                  OUT.01: .WORD 0                 ;DATA ARGUMENT  1
001502  000000                  OUT.02: .WORD 0                 ;DATA ARGUMENT  2
001503  000000                  OUT.03: .WORD 0                 ;DATA ARGUMENT  3
001504  000000                  OUT.04: .WORD 0                 ;DATA ARGUMENT  4
001505  000000                  OUT.05: .WORD 0                 ;DATA ARGUMENT  5
001506  000000                  OUT.06: .WORD 0                 ;DATA ARGUMENT  6
001507  000000                  OUT.07: .WORD 0                 ;DATA ARGUMENT  7
001510  000000                  OUT.08: .WORD 0                 ;DATA ARGUMENT  8
001511  000000                  OUT.09: .WORD 0                 ;DATA ARGUMENT  9
001512  000000                  OUT.10: .WORD 0                 ;DATA ARGUMENT 10
001513  000000                  OUT.11: .WORD 0                 ;DATA ARGUMENT 11
001514  000000                  OUT.12: .WORD 0                 ;DATA ARGUMENT 12
001515  000000                  OUT.13: .WORD 0                 ;DATA ARGUMENT 13
001516  000000                  OUT.14: .WORD 0                 ;DATA ARGUMENT 14
001517  000000                  OUT.15: .WORD 0                 ;DATA ARGUMENT 15
001520  000000                  OUT.16: .WORD 0                 ;DATA ARGUMENT 16
001521  000000                  OUT.17: .WORD 0                 ;DATA ARGUMENT 17
001522  000000                  OUT.18: .WORD 0                 ;DATA ARGUMENT 18
001523  000000                  OUT.19: .WORD 0                 ;DATA ARGUMENT 19
001524  000000                  OUT.20: .WORD 0                 ;DATA ARGUMENT 20
001525  000000                  OUT.21: .WORD 0                 ;DATA ARGUMENT 21
001526  000000                  OUT.22: .WORD 0                 ;DATA ARGUMENT 22
001527  000000                  OUT.23: .WORD 0                 ;DATA ARGUMENT 23
001530  000000                  OUT.24: .WORD 0                 ;DATA ARGUMENT 24
001531  000000                  OUT.25: .WORD 0                 ;DATA ARGUMENT 25
001532  000000                  OUT.26: .WORD 0                 ;DATA ARGUMENT 26
001533  000000                  OUT.27: .WORD 0                 ;DATA ARGUMENT 27
001534  000000                  OUT.28: .WORD 0                 ;DATA ARGUMENT 28
001535  000000                  OUT.29: .WORD 0                 ;DATA ARGUMENT 29
001536  000000                  OUT.30: .WORD 0                 ;DATA ARGUMENT 30
001537  000000                  OUT.31: .WORD 0                 ;DATA ARGUMENT 31
001540  000000                  OUT.32: .WORD 0                 ;DATA ARGUMENT 32
001541  000000                  OUT.33: .WORD 0                 ;DATA ARGUMENT 33
001542  000000                  OUT.34: .WORD 0                 ;DATA ARGUMENT 34
                                ;
                                ;IN BUFFER - DATA RECEIVED FROM HOST
                                ;
001543  000000                  IN.RQ:  .WORD 0                 ;HOST REQUEST CODE (ECHO)
001544  000000                  IN.01:  .WORD 0                 ;DATA ARGUMENT  1
001545  000000                  IN.02:  .WORD 0                 ;DATA ARGUMENT  2
001546  000000                  IN.03:  .WORD 0                 ;DATA ARGUMENT  3
001547  000000                  IN.04:  .WORD 0                 ;DATA ARGUMENT  4
001550  000000                  IN.05:  .WORD 0                 ;DATA ARGUMENT  5
001551  000000                  IN.06:  .WORD 0                 ;DATA ARGUMENT  6
001552  000000                  IN.07:  .WORD 0                 ;DATA ARGUMENT  7
001553  000000                  IN.08:  .WORD 0                 ;DATA ARGUMENT  8
001554  000000                  IN.09:  .WORD 0                 ;DATA ARGUMENT  9
001555  000000                  IN.10:  .WORD 0                 ;DATA ARGUMENT 10
001556  000000                  IN.11:  .WORD 0                 ;DATA ARGUMENT 11
001557  000000                  IN.12:  .WORD 0                 ;DATA ARGUMENT 12
001560  000000                  IN.13:  .WORD 0                 ;DATA ARGUMENT 13
```

```
     58 001561  000000                IN.14:  .WORD 0                 ;DATA ARGUMENT 14
     59 001562  000000                IN.15:  .WORD 0                 ;DATA ARGUMENT 15
     60 001563  000000                IN.16:  .WORD 0                 ;DATA ARGUMENT 16
     61 001564  000000                IN.17:  .WORD 0                 ;DATA ARGUMENT 17
     62 001565  000000                IN.18:  .WORD 0                 ;DATA ARGUMENT 18
     63 001566  000000                IN.19:  .WORD 0                 ;DATA ARGUMENT 19
     64 001567  000000                IN.20:  .WORD 0                 ;DATA ARGUMENT 20
     65 001570  000000                IN.21:  .WORD 0                 ;DATA ARGUMENT 21
     66 001571  000000                IN.22:  .WORD 0                 ;DATA ARGUMENT 22
     67 001572  000000                IN.23:  .WORD 0                 ;DATA ARGUMENT 23
     68 001573  000000                IN.24:  .WORD 0                 ;DATA ARGUMENT 24
     69 001574  000000                IN.25:  .WORD 0                 ;DATA ARGUMENT 25
     70 001575  000000                IN.26:  .WORD 0                 ;DATA ARGUMENT 26
     71 001576  000000                IN.27:  .WORD 0                 ;DATA ARGUMENT 27
     72 001577  000000                IN.28:  .WORD 0                 ;DATA ARGUMENT 28
     73 001600  000000                IN.29:  .WORD 0                 ;DATA ARGUMENT 29
     74 001601  000000                IN.30:  .WORD 0                 ;DATA ARGUMENT 30
     75 001602  000000                IN.31:  .WORD 0                 ;DATA ARGUMENT 31
     76 001603  000000                IN.32:  .WORD 0                 ;DATA ARGUMENT 32
     77 001604  000000                IN.33:  .WORD 0                 ;DATA ARGUMENT 33
     78 001605  000000                IN.34:  .WORD 0                 ;DATA ARGUMENT 34
     79         000043                BUFSIZ  =       . - IN.RQ       ;SIZE OF BUFFER
```

```
                                        .SBTTL  STACK AREA
                                ;STACK AREA

                                        .WORD 123456            ;END MARKER FOR STACK
                                        .BLKW 31.               ;STACK
                                STACK:  .WORD 123456            ;MARKER FOR STACK UNDERFLOW
```

```
                                        .SBTTL  GETU   - POLL ALL PORTS, THEN GET UNITS TO TEST

                                        ;GET WHICH UNITS TO TEST

001647                                  START:
                                        ;
                                        ;       GET THE UNITS TO TEST
                                        ;
                                        ;
                                        ;       POLL ALL PORTS AND FILL IN A UDA PORT INFORMATION TABLE (UNITS)
                                        ;
001647  104205  000001                          MOV     #1,R5           ; MOVE INITIAL MASK TO R5
001651  104204  000720                          MOV     #UNITS,R4       ; R4 POINTS TO UNIT TABLE
001653                                  5$:     PUSH    R4              ; SAVE R4
001653  100464                                                                  MOV R4,-(SP)
001654  104052                          MOV     R5,R2           ; MOVE MASK TO R2
001655  024612                          CALL    RDSTAT          ; GET DRIVE'S STATUS
001656  102201  010000                  BIT     #10000,R1       ; SEE IF ERROR
001660  011666                          BEQ     10$             ; IF NOT, BRANCH
001661  104203  003772                  MOV     #SER10,R3       ; NO DRIVE ATTACHED
001663  100643  000001                  MOV     R3,1(R4)        ; SAVE ERROR MESSAGE
001665  002056                          BR      85$             ; REPORT
001666  114003                  10$:    CLR     R3              ; SET UP TIMEOUT COUNT
001667  024612                  15$:    CALL    RDSTAT          ; GET STATUS
001670  102201  000001                  BIT     #RCVRDY,R1      ; SEE IF RECEIVER READY ASSERTED
001672  051702                          BNE     20$             ; IF SO, BRANCH
001673  117403                          DEC     R3              ; DECREMENT COUNT
001674  051667                          BNE     15$             ; IF INCOMPLETE, BRANCH
001675  104203  004005                  MOV     #SER11,R3       ; RECEIVER READY NEVER ASSERTED
001677  100643  000001                  MOV     R3,1(R4)        ; SAVE ERROR MESSAGE
001701  002056                          BR      85$             ; REPORT
001702  102201  000100          20$:    BIT     #AVAIL,R1       ; SEE IF DRIVE IS AVAILABLE
001704  051712                          BNE     25$             ; IF SO, BRANCH
001705  104203  004215                  MOV     #SER40,R3       ; GET SECONDARY ERROR
001707  100643  000001                  MOV     R3,1(R4)        ; SAVE
001711  002056                          BR      85$             ; EXIT
001712  104202  001750          25$:    MOV     #MAXSND,R2      ; SET UP MAXIMUM TRIES AT SENDING
001714  104203  001170                  MOV     #CR.GST,R3      ; R3 POINTS TO GET STATUS COMMAND
001716  104137                  30$:    MOV     (R3),R0         ; SET ADR OF SDI COMMAND BUFFER
001717  104631  000001                  MOV     1(R3),R1        ; SET BUFFER LENGTH
001721                                  PUSH    R2              ; SAVE R2
001721  100462                                                                  MOV R2,-(SP)
001722  104052                          MOV     R5,R2           ; SETUP FOR SEND
001723  060004                          XFC     SEND            ; SEND COMMAND
001724                                  POP     R2              ; RESTORE COUNT
001724  104262                                                                  MOV (SP)+,R2
001725  115001                          TST     R1              ; DID UNIT ACCEPT COMMAND
001726  011736                          BEQ     35$             ; IF SO, BRANCH
001727  117402                          DEC     R2              ; DECREMENT COUNT
001730  051716                          BNE     30$             ; IF UNEXPIRED, BRANCH
001731  104203  004023                  MOV     #SER12,R3       ; GET ERROR NUMBER
001733  100643  000001                  MOV     R3,1(R4)        ; SAVE
001735  002056                          BR      85$
001736                                  PUSH    R4              ; SAVE R4
001736  100464                                                                  MOV R4,-(SP)
001737  104204  000003                  MOV     #3,R4           ; SET UP SHORT TIMEOUT
```

```
     54 001741  104637  000002          40$:    MOV     2(R3),R0        ; SET DATA BUFFER ADDRESS
     55 001743  104631  000003                  MOV     3(R3),R1        ; SET BUFFER LENGTH
     56 001745  104052                          MOV     R5,R2           ; SETUP FOR RECEIVE
     57 001746  060005                          XFC     RCV             ; RECEIVE SDI COMMAND
     58 001747  115001                          TST     R1              ; DID ERROR OCCUR
     59 001750  012010                          BEQ     70$             ; IF NOT, BRANCH
     60 001751  106201  000001                  CMP     #1,R1           ; SEE IF TIMEOUT
     61 001753  051762                          BNE     45$             ; IF NOT, BRANCH
     62 001754  117404                          DEC     R4              ; DECREMENT TIMEOUT VALUE
     63 001755  051741                          BNE     40$             ; IF NOT TIMEOUT, BRANCH
     64 001756                                  POP     R4              ; RESTORE R4
        001756  104264                                                                  MOV (SP)+,R4
     65 001757  104203  004035                  MOV     #SER13,R3       ; GET ERROR NUMBER
     66 001761  002005                          BR      65$             ; BRANCH TO EXIT
     67 001762                          45$:    POP     R4              ; RESTORE R4
        001762  104264                                                                  MOV (SP)+,R4
     68 001763  110601                          ROR     R1              ; ROTATE INTO POSITION TO TEST
     69 001764  110601                          ROR     R1              ; SEE IF FIRST WORD NOT START FRAME
     70 001765  041771                          BCC     50$             ; IF NOT, BRANCH
     71 001766  104203  004050                  MOV     #SER14,R3       ; GET ERROR NUMBER
     72 001770  002005                          BR      65$             ; BRANCH TO END OF LOOP
     73 001771  110601                  50$:    ROR     R1              ; SEE IF FRAMING ERROR
     74 001772  041776                          BCC     55$             ; IF NOT, BRANCH
     75 001773  104203  004076                  MOV     #SER15,R3       ; GET ERROR NUMBER
     76 001775  002005                          BR      65$             ; BRANCH TO END OF LOOP
     77 001776  110601                  55$:    ROR     R1              ; SEE IF CHECKSUM ERROR
     78 001777  042003                          BCC     60$             ; IF NOT, BRANCH
     79 002000  104203  004120                  MOV     #SER16,R3       ; GET ERROR NUMBER
     80 002002  002005                          BR      65$             ; BRANCH TO END OF LOOP
     81 002003  104203  004143          60$:    MOV     #SER17,R3       ; GET ERROR NUMBER
     82 002005  100643  000001          65$:    MOV     R3,1(R4)        ; SAVE
     83 002007  002056                          BR      85$             ; BRANCH TO END OF LOOP
```

```
                                        ;
                                        ;       NOW FILL IN THE TABLE WITH ALL THE SUBUNIT NUMBERS
                                        ;
002010                          70$:    POP     R4              ; RESTORE R4
002010  104264                                                                  MOV (SP)+,R4
002011  104207  177777                  MOV     #-1,R0          ; GET 'NO UNITS' FLAG
002013  100647  000001                  MOV     R0,1(R4)        ; CLEAR ANY ERRORS THAT ARE FLAGGED
002015  100647  000002                  MOV     R0,2(R4)        ; CLEAR ANY ERRORS THAT ARE FLAGGED
002017  104307  000744                  MOV     ST,R0           ; R0 HAS UNIT NUMBER
002021  104072                          MOV     R0,R2           ; COPY R0 TO R2
002022  103207  170000                  BIC     #^CHBHINB,R0    ; R0 HAS UNIT NUMBER
002024                                  PUSH    <R1,R2>         ; SAVE R1 AND R2
002024  100461                                                                  MOV R1,-(SP)
002025  100462                                                                  MOV R2,-(SP)
002026  104052                          MOV     R5,R2           ; MOVE UDA PORT MASK TO R2
002027  024612                          CALL    RDSTAT          ; GET STATUS
002030  102201  000002                  BIT     #ATTN,R1        ; SEE IF SPINABLE
002032  052035                          BNE     75$             ; IF SO, BRANCH
002033  101207  010000                  BIS     #10000,R0       ; SET 'NOT SPINABLE' FLAG
002035                          75$:    POP     <R2,R1>         ; RESTORE
002035  104262                                                                  MOV (SP)+,R2
002036  104261                                                                  MOV (SP)+,R1
002037  101207  040000                  BIS     #40000,R0       ; FLAG UNIT AS NOT TESTED
002041  110702                          SWAB    R2              ; SWAP R2'S BYTES
002042  110602                          ROR     R2              ; MOVE SUBUNIT MASK TO LO NIBBLE
002043  110602                          ROR     R2              ; MOVE SUBUNIT MASK TO LO NIBBLE
002044  110602                          ROR     R2              ; MOVE SUBUNIT MASK TO LO NIBBLE
002045  110602                          ROR     R2              ; MOVE SUBUNIT MASK TO LO NIBBLE
002046  103202  177760                  BIC     #LBLONB,R2      ; CLEAR ALL BUT SUBUNIT BITS
002050  100147                          MOV     R0,(R4)         ; MOVE UNIT NUMBER INTO UNITS
002051  110602                  80$:    ROR     R2              ; MOVE SUBUNIT BIT TO CARRY
002052  042056                          BCC     85$             ; IF NO MORE SUBUNITS, BRANCH
002053  100247                          MOV     R0,(R4)+        ; MOVE SUBUNIT NUMBER TO TABLE
002054  115407                          INC     R0              ; INCREMENT SUBUNIT NUMBER
002055  002051                          BR      80$             ; BRANCH
                                        ;
                                        ;
                                        ;
002056                          85$:    POP     R4              ; R4 POINTS TO START OF UNIT JUST HANDLED
002056  104264                                                                  MOV (SP)+,R4
002057  105204  000004                  ADD     #4,R4           ; R4 WILL POINT TO NEXT UNIT (CLEAR CARRY FOR ROL)
002061  110205                          ROL     R5              ; R5 HAS NEXT UNIT PORT MASK
002062  106205  000020                  CMP     #20,R5          ; SEE IF ALL PORTS TESTED
002064  051653                          BNE     5$              ; IF NOT, BRANCH
```

```
                                        ;
                                        ;       NOW GET THE PLUG NUMBERS TO TEST AND FIND THEM IN THE TABLE
                                        ;
002065  104207  060012                  MOV     #UTOTST,R0      ; GET WHAT SUBUNIT NUMBERS TO TEST REQUEST
002067  024643                          CALL    HOSTRQ          ; GET THE PLUG NUMBERS
002070  104207  001544                  MOV     #IN.01,R0       ; R0 POINTS TO UNIT NUMBERS TO TEST
002072  104201  000720          90$:    MOV     #UNITS,R1       ; R1 POINTS TO UDA PORT INFORMATION
002074  104112                  95$:    MOV     (R1),R2         ; R2 HAS UNIT
002075  072120                          BMI     115$            ; IF NONE, BRANCH
002076                                  PUSH    R1              ; SAVE R1
002076  100461                                                                  MOV R1,-(SP)
002077  103202  170000          100$:   BIC     #170000,R2      ; CLEAR 'NOT TESTED', DO NOT LEAVE 'UNSPINABLE' SET
002101  106172                          CMP     (R0),R2         ; SEE IF IT IS A UNIT TO TEST
002102  052106                          BNE     105$            ; NO MATCH
002103  100112                          MOV     R2,(R1)         ; SAVE UNIT AS ONE TO TEST
002104                                  POP     R1              ; RESTORE STACK
002104  104261                                                                  MOV (SP)+,R1
002105  002225                          BR      160$            ; LOOK FOR THE NEXT ONE
002106  115401                  105$:   INC     R1              ; POINT TO NEXT SUBUNIT
002107  104012                          MOV     R1,R2           ; COPY TO R2
002110  107202  000720                  SUB     #UNITS,R2       ; SUBTRACT STARTING ADDRESS
002112  102202  000003                  BIT     #3,R2           ; SEE IF STILL ON SAME UNIT
002114  012117                          BEQ     110$            ; IF NOT, BRANCH
002115  104112                          MOV     (R1),R2         ; SEE IF ANY MORE SUBUNITS
002116  032077                          BPL     100$            ; IF SO, BRANCH
002117                          110$:   POP     R1              ; RESTORE R1
002117  104261                                                                  MOV (SP)+,R1
002120  105201  000004          115$:   ADD     #4,R1           ; LOOK AT NEXT UNIT
002122  106201  000740                  CMP     #UNITS+16.,R1   ; SEE IF ENTIRE TABLE SEARCHED
002124  052074                          BNE     95$             ; IF NOT, BRANCH
```

```
                                        ;
                                        ;       DIDN'T FIND THE REQUESTED UNITS -- DUMP ALL KNOWLEDGE OF THE
                                        ;       UDA PORTS AND DIE
                                        ;
002125  104170  001503                  MOV     (R0),OUT.03     ; SAVE UNIT NUMBER IN REQUEST BUFFER
002127  104204  001505                  MOV     #OUT.05,R4      ; R4 POINTS TO OUTPUT BUFFER
002131  104205  000720                  MOV     #UNITS,R5       ; R5 POINTS TO UNIT TABLE
002133  104157                  120$:   MOV     (R5),R0         ; GET FIRST WORD OF UNIT
002134  032141                          BPL     125$            ; IF VALID UNIT WAS FOUND, BRANCH
002135  104657  000001                  MOV     1(R5),R0        ; GET POINTER TO ERROR MESSAGE
002137  100247                          MOV     R0,(R4)+        ; SAVE IN OUTPUT BUFFER
002140  002174                          BR      155$            ; EXIT
002141                  125$:   PUSH    R5                      ; SAVE POINTER TO UNIT TABLE
002141  100465                                                                  MOV R5,-(SP)
002142  102207  010000                  BIT     #10000,R0       ; SEE IF DRIVE UNSPINABLE
002144  012150                          BEQ     130$            ; IF SPINABLE, BRANCH
002145  104207  004237                  MOV     #SER41,R0       ; REPORT DRIVE(S) UNSPINABLE
002147  002152                          BR      135$            ; BRANCH
002150  104207  004170          130$:   MOV     #SER18,R0       ; GET POINTER TO ERROR MESSAGE
002152  100247                  135$:   MOV     R0,(R4)+        ; SAVE
002153  114007                          CLR     R0              ; CLEAR COUNT
002154  104251                  140$:   MOV     (R5)+,R1        ; GET UNIT NUMBER
002155  072162                          BMI     145$            ; IF INVALID, BRANCH
002156  115407                          INC     R0              ; INCREMENT COUNT
002157  106207  000004                  CMP     #4,R0           ; SEE IF MAX
002161  052154                          BNE     140$            ; IF NOT, LOOP
002162  104671  001155          145$:   MOV     SER18E-1(R0),R1 ; GET POINTER TO CORRECT ERROR MESSAGE
002164  100241                          MOV     R1,(R4)+        ; MOVE INTO OUTPUT BUFFER
002165                                  POP     R5              ; RESTORE R5
002165  104265                                                                  MOV (SP)+,R5
002166                                  PUSH    R5              ; SAVE R5
002166  100465                                                                  MOV R5,-(SP)
002167  104251                  150$:   MOV     (R5)+,R1        ; GET SUBUNIT NUMBER
002170  100241                          MOV     R1,(R4)+        ; SAVE
002171  117407                          DEC     R0              ; DECREMENT COUNT
002172  052167                          BNE     150$            ; IF COUNT INCOMPLETE, BRANCH
002173                                  POP     R5              ; RESTORE R5
002173  104265                                                                  MOV (SP)+,R5
002174  105205  000004          155$:   ADD     #4,R5           ; POINT TO NEXT UNIT TABLE
002176  106205  000740                  CMP     #UNITS+16.,R5   ; SEE IF ENTIRE TABLE SEARCHED
002200  052133                          BNE     120$            ; IF NOT, BRANCH
002201                                  DEVFTL  5000            ; REPORT FATAL ERROR (WILL SHOW UP AS A 5000)
002201  104200  003653  001504                                                  MOV     #ER5000,OUT.04
002204  104202  051610                                                          MOV     #5000!FTLDEV,R2
002206  104020  001502                                                          MOV     R2,OUT.02
002210  104200  002210  001501                                                  MOV     #.,OUT.01
002213  104200  060014  001500                                                  MOV     #ERRMC,OUT.RQ
002216  104307  001500                  MOV     OUT.RQ,R0       ; SET UP FOR REPORT
002220  024643                          CALL    HOSTRQ          ; REPORT ERROR
002221  104207  060016                  MOV     #DONE,R0        ; END TEST
002223  024643                          CALL    HOSTRQ          ; SEND END-OF-TEST TO HOST
002224  060021                          XFC     EXIT            ; TERMINATE DM

002225  115407          160$:   INC     R0              ; POINT TO NEXT UNIT TO TEST
002226  104172                  MOV     (R0),R2         ; CHECK NEXT UNIT
002227  032072                  BPL     90$             ; FIND IN UDA PORT INFORMATION
```

```
                                                .SBTTL  TEST 2 START TESTING
                                        ;:+
                                        ;       FIND OUT IF HOST HAD TO INIT UDA TO GET A PROGRAM FROM AN ATTACHED DRIVE.
                                        ;       THE PROGRAM WAS REQUESTED BY THE DRIVE TO BE DOWNLINE LOADED INTO ITS MEMORY.
                                        ;       THE INDICATION COMES FROM THE UTOTST RESPONSE.
                                        ;
                                        ;       A WORD WITH MSB IS SET TO INDICATE THE LAST UNIT ENTRY.  AFTER THAT WORD
                                        ;       ANOTHER WORD INDICATES WHEATHER OR NOT A FILE WAS SUPPOSE TO BE DOWN LINE LOADED.
                                        ;       A '0' MEANS NORMAL PROCESSING, NO FILE WAS EXPECTED OR REQUESTED.
                                        ;       A '1' MEANS A FILE WAS EXPECTED AND IS AVAILABLE.
                                        ;       A '2' MEANS A FILE WAS EXPECTED AND IS NOT AVAILABLE.
                                        ;
                                        ;       R0 IS A POINTER TO THE WORD BEYOND THE LAST TABLE ENTRY
                                        ;       (R0) = 100000
                                        ;:-
002230  104671  000001        T2STRT: MOV     1(R0),R1                ;WAS A FILE EXPECTED?
002232  012314                        BEQ     PORT0                   ;IF NOT, GO TO NORMAL PROCESSING
002233  115407                        INC     R0                      ;POINT TO INDICATOR IN IN.RQ BUFFER
002234  104075                        MOV     R0,R5                   ;R5 -> INDICATOR
002235  104151                        MOV     (R5),R1                 ;1ST WORD CONTAINS INDICATOR TO SEE IF A FILE WAS EX
002236  106201  000001                CMP     #1,R1                   ;WAS THE EXPECTED FILE AVAILABLE?
002240  053525                        BNE     D0.DI7                  ;IF NOT AVAILABLE, GO TO REPORT ERROR
                                ; *** FILE WAS THERE, NOW SEE WHERE IN THE TABLE IT WAS SO PORT INDICATOR IS THE SAME?
002241  104207  000720                MOV     #UNITS,R0               ;R0 -> UNIT TABLE
002243  104651  000001                MOV     1(R5),R1                ;R1 = UNIT NUMBER
002245  104202  000001                MOV     #UNIT0,R2               ;R2 = 1ST PORT
002247  114003                        CLR     R3                      ;R3 = UNIT TABLE INDICATOR
002250  114004                1$:     CLR     R4                      ;R4 KEEPS TRACK OF WHICH SUBUNITS
002251  104270  001153        2$:     MOV     (R0)+,SAVREG            ;STORE UNIT IN SAVE REG
002253  072300                        BMI     4$                      ;IF NEGATIVE VALUE, DON'T COMPARE
002254  103300  177400  001153        BIC     HIBYTE,SAVREG           ;CLEAR EXTANEOUS BITS
002257  106301  001153                CMP     SAVREG,R1               ;DID WE FIND THE DRIVE?
002261  012304                        BEQ     5$                      ;IF IT IS, WE HAVE FOUND IT
002262  115404                        INC     R4                      ;ELSE, INCREMENT SUBUNIT POINTER
002263  106204  000004                CMP     #4,R4                   ;ENDED WITH THIS UNIT ENTRY?
002265  052251                        BNE     2$                      ;IF NOT, CONTINUE
002266  105022                3$:     ADD     R2,R2                   ;ELSE, NEXT PORT SET IN R2
002267  115403                        INC     R3                      ;INCREMENT UNIT TABLE INDICATOR
002270  106203  000004                CMP     #4,R3                   ;DONE?
002272  052250                        BNE     1$                      ;IF THIS GETS TO 4, THEN ERROR(UNIT NOT FOUND)
002273                                ERRHRD  MS2000,R1
                                                                                MOV R1,-(SP)
                                                                                CALL RERROR     ;ERROR # 2000.
002273  100461                                                                  .WORD ERHARD+ERRN
002274  025016                                                                  .WORD <PRMS*10000>+MS2000
002275  103720
002276  010000
002277  002314                        BR      PORT0                   ;START REGULAR TESTING
                                ; *** IF HERE, NEGATIVE ENTRY, GO TO NEXT UNIT ENTRY.  ADJUST R0 TO PROPER POINTER
002300  105207  000003        4$:     ADD     #3,R0                   ;R0 -> INTO NEXT UNIT\INCREMENTED BY (R0)+
002302  107047                        SUB     R4,R0                   ;REALIGN TO POINT TO BEGINNING OF POINTER
002303  002266                        BR      3$                      ;GO BACK INTO THE LOOP
                                ; *** IF HERE, FOUND THE ENTRY, SAVE PERTINENT VALUES, GO DOWN LINE LOAD PROGRAM
002304  104650  000001  001152  5$:     MOV     1(R5),LUNIT             ;SAVE LOGICAL UNIT NUMBER
002307  104020  000741                MOV     R2,SDI                  ;SAVE PORT VALUE
002311  104030  000740                MOV     R3,UNITNB               ;SAVE UNIT NUMBER OFFSET IN TABLE
002313  003370                        BR      D0.DI2                  ;GO DOWN LINE LOAD PROGRAM
```

```
                                        .SBTTL          THE DIAGNOSTICS TO ALL UNITS SELECTED
                                ;         THE DIAGNOSTICS TO ALL UNITS SELECTED.
                                ;TEST CODE WILL BE CALLED FOR EACH DISK SUBUNIT TO BE TESTED.
                                ;  LUNIT WILL CONTAIN LOGICAL UNIT NUMBER OF DRIVE FOR ERROR REPORTS
                                ;  SDI WILL CONTAIN SDI INTERCONNECT CODE FOR SELECTED DRIVE
                                ;  UNITNB WILL CONTAIN AN EVEN NUMBER FOR TESTING FIRST SUBUNIT OF A DRIVE
                                ;                      AN ODD NUMBER FOR TESTING SECOND SUBUNIT OF A DRIVE

                                ; *** NORMAL PROCESSING
002314                          PORT0:
002314  114001                         CLR     R1              ;START WITH UNIT 0 INDEX

002315  104207  000720         PORT2:  MOV     #UNITS,R0       ; GET POINTER TO UNITS TABLE
002317  105017                         ADD     R1,R0           ; ADD INDEX
002320  104173                         MOV     (R0),R3         ; GET CONTENTS OF TABLE
002321  032330                         BPL     1$              ; IF THIS UNIT IS PRESENT, BRANCH
002322  102201  000003                 BIT     #3,R1           ; SEE IF ON SUBUNIT 0 OF UNIT
002324  052362                         BNE     PORT5           ; IF NOT, TEST NEXT SUBUNIT
002325  105201  000003                 ADD     #3,R1           ; IF NO SUBUNIT 0, THEN NO UNIT - SKIP OTHER SUBUNITS
002327  002362                         BR      PORT5           ; BYPASS IF NO UNIT
002330  102203  040000         1$:     BIT     #40000,R3       ; SEE IF THIS UNIT IS TO BE TESTED
002332  052362                         BNE     PORT5           ; IF NOT, BRANCH
002333  104010  000740                 MOV     R1,UNITNB       ; STORE UNIT INDEX
002335  104030  001152                 MOV     R3,LUNIT        ; STORE LOGICAL UNIT NUMBER FOR DRIVE
002337  104202  000001                 MOV     #UNIT0,R2       ; GET UNIT 0 INTERCONNECT CODE
002341  110601                         ROR     R1              ; DIVIDE UNITNB BY FOUR
002342  110601                         ROR     R1
002343  103201  177760                 BIC     #LBLONB,R1      ; CLEAR UNUSED BITS
002345  117401         PORT3:  DEC     R1
002346  072353                         BMI     PORT4           ; FOR EACH DRIVE OVER 0 SHIFT R2 LEFT
002347  110202                         ROL     R2
002350  103202  000001                 BIC     #1,R2           ; CLEAR CARRY ROTATED INTO REG (IF ANY)
002352  002345                         BR      PORT3
002353  104020  000741         PORT4:  MOV     R2,SDI          ; STORE SDI INTERCONNECT CODE
002355  002372                         BR      TEST            ; PERFORM TEST ON THIS DRIVE
                                                                ; TEST RETURNS TO TESTX
002356  104206  001646         TESTX:  MOV     #STACK,SP       ; RESET STACK DO TO JUMPS OUT OF
                                                                ; SUBROUTINES
002360  104301  000740                 MOV     UNITNB,R1       ; GET UNIT INDEX
002362  115401         PORT5:  INC     R1              ; INCREMENT INDEX
002363  106201  000020                 CMP     #16.,R1         ; CHECK IF 16 DRIVES ALREADY SELECTED
002365  052315                         BNE     PORT2           ; REPEAT FOR ALL DRIVES
002366  104207  060016         DONECD: MOV     #DONE,R0        ; END OF PROGRAM
002370  024643                         CALL    HOSTRQ
002371  002366                         BR      DONECD          ; REPEAT IF RETURNED
```

INITIALIZE DRIVE AND LOOK AT DRIVE SIGNALS

```
                                .SBTTL  INITIALIZE DRIVE AND LOOK AT DRIVE SIGNALS

                                ;START OF TEST CODE
                                ;
                                ;INPUTS:
                                ;       LUNIT - LOGICAL UNIT NUMBER OF DRIVE UNDER TEST
                                ;       SDI - SDI INTERCONNECT CODE FOR DRIVE

                                ;INITIALIZE THE DRIVE

002372  114003                  TEST:   CLR     R3                      ;R3 IS A DECREMENT COUNTER TO CHECK IF THE
                                                                        ; COMMAND HAS BEEN RECEIVED BY THE DRIVE
002373  104302  000741                  MOV     SDI,R2                  ;GET SDI SELECT CODE
002375  060011                          XFC     DINIT                   ;INITIALIZE THE DRIVE

                                ;WAIT FOR DRIVE TO ASSERT RECEIVER READY
                                ;TIME OUT AFTER ...?

002376  114005                          CLR     R5                      ;GET TIMEOUT COUNTER
002377  024612                  ILOOP:  CALL    RDSTAT                  ;GET DRIVE STATUS
002400  102201  010000                  BIT     #10000,R1               ;SEE IF ANY ERRORS
002402  012417                          BEQ     2$                      ;IF NO ERROR, BRANCH
002403  102201  004000                  BIT     #4000,R1                ;SEE IF XMIT ERROR
002405  117403                          DEC     R3                      ;ON SEND ERROR, DID WE DEPLETE THE COUNTER?
002406  052377                          BNE     ILOOP                   ;IF NOT, TRY AGAIN
002407                                  ERRHRD  MS2001                  ;REPORT INVALID STATUS ERROR
002407  025016                                                                  CALL RERROR     ;ERROR # 2001.
002410  103721                                                                  .WORD ERHARD+ERRN
002411  000043                                                                  .WORD <PRMS*10000>+MS2001
002412  002366                          BR      DONECD                  ;BRANCH TO DONE
002413                          1$:     ERRHRD  MS2002                  ;REPORT XMIT ERROR
002413  025016                                                                  CALL RERROR     ;ERROR # 2002.
002414  103722                                                                  .WORD ERHARD+ERRN
002415  000126                                                                  .WORD <PRMS*10000>+MS2002
002416  002366                          BR      DONECD                  ;BRANCH TO DONE
002417                          2$:
002417  114003                          CLR     R3                      ;R3 IS A DECREMENT COUNTER TO CHECK IF THE
                                                                        ; COMMAND HAS BEEN RECEIVED BY THE DRIVE
002420  102201  000001                  BIT     #RCVRDY,R1              ;CHECK RECEIVER READY LINE
002422  052430                          BNE     ECHO1                   ;ADVANCE IF ASSERTED
002423  117405                          DEC     R5                      ;DECREMENT TIME OUT COUNTER
002424  052377                          BNE     ILOOP                   ;STAY IN LOOP UNTIL SIGNAL SETS OR TIMEOUT
002425                                  ERRHRD  MS2003                  ;REPORT ERROR
002425  025016                                                                  CALL RERROR     ;ERROR # 2003.
002426  103723                                                                  .WORD ERHARD+ERRN
002427  000165                                                                  .WORD <PRMS*10000>+MS2003
```

```
                                .SBTTL  ECHO DATA TO DRIVE

                                ;ECHO THE FOLLOWING DATA PATTERNS TO THE DRIVE THEN CHECK THE
                                ;RESPONSE FOR THE PROPER DATA RECEIVED:
                                ;       377
                                ;       000
                                ;       252
                                ;       360
                                ;       017

002430  104205  001162      ECHO1:  MOV     #ECHOD,R5               ;GET POINTER TO ECHO DATA
002432  104157              ECHO1A: MOV     (R5),R0                 ;GET ECHO DATA
002433  103207  177400      ECHO2:  BIC     #HIBYTE,R0              ;CLEAR COMMAND FROM WORD
002435  060010                      XFC     ECHO                    ;PERFORM THE ECHO COMMAND

                                ;CHECK FOR TIMEOUT ERROR

002436  115001                      TST     R1                      ;CHECK FOR ERROR
002437  012457                      BEQ     ECHO4                   ;BRANCH IF NONE
002440  102201  000001              BIT     #1,R1                   ;CHECK IF SEND ERROR
002442  012452                      BEQ     ECHO3                   ;BRANCH IF RECEIVE ERROR
002443  117403                      DEC     R3                      ;IF SEND ERROR, IS THE COUNTER BEEN DEPLETED
002444  052432                      BNE     ECHO1A                  ;IF NOT, TRY AGAIN
002445                              ERRHRD  MS2004,R0               ;REPORT SEND ERROR
002445  100467                                                              MOV R0,-(SP)
002446  025016                                                              CALL RERROR     ;ERROR # 2004.
002447  103724                                                              .WORD ERHARD+ERRN
002450  010235                                                              .WORD <PRMS*10000>+MS2004
002451  002473                      BR      ECHO5
002452                      ECHO3:  ERRHRD  MS2005,R0               ;REPORT RECEIVE ERROR
002452  100467                                                              MOV R0,-(SP)
002453  025016                                                              CALL RERROR     ;ERROR # 2005.
002454  103725                                                              .WORD ERHARD+ERRN
002455  010275                                                              .WORD <PRMS*10000>+MS2005
002456  002473                      BR      ECHO5

                                ;CHECK DATA RECEIVED

002457  106157              ECHO4:  CMP     (R5),R0                 ;COMPARE DATA RECEIVED WITH
002460  012473                      BEQ     ECHO5                   ; DATA SENT
002461                              ERRHRD  MS2006,(R5),R0          ;REPORT DATA COMPARE ERROR
002461  100467                                                              MOV R0,-(SP)
002462  104010  001153                                                      MOV R1,SAVREG
002464  104151                                                              MOV (R5),R1
002465  100461                                                              MOV R1,-(SP)
002466  104301  001153                                                      MOV SAVREG,R1
002470  025016                                                              CALL RERROR     ;ERROR # 2006.
002471  103726                                                              .WORD ERHARD+ERRN
002472  020340                                                              .WORD <PRMS*10000>+MS2006

                                ;MOVE TO NEXT DATA PATTERN

002473  114003              ECHO5:  CLR     R3
002474  115405                      INC     R5                      ;BUMP TO NEXT DATA TO SEND
002475  104157                      MOV     (R5),R0                 ;CHECK IF AT END OF TABLE
002476  072433                      BMI     ECHO2                   ;SEND THIS DATA IF NOT
```

```
                                .SBTTL  GET STATUS COMMAND
                                ;
                                ;ISSUE GET STATUS COMMAND AND CHECK THAT IT PERFORMS PROPERLY
                                ;
002477  104200  000002  000742          MOV     #2,SAVSTA               ;COUNTER
002502  024131                          CALL    GTSTAT                  ;GET STATUS
                                ;
                                ;CLEAR DRIVE ERRORS
                                ;
002503  024211                DRCLR1: CALL    CLRDRV                  ; CLEAR DRIVE
                                ;
                                ;
                                ;ISSUE GET STATUS COMMAND AND CHECK THAT IT PERFORMS PROPERLY
                                ;
002504                          GSTS3:
002504  024131                          CALL    GTSTAT                  ;GET STATUS
                                ;
                                ;LOOK AT DATA IN RESPONSE PACKET
                                ;
002505  104307  000745          GSTS6:  MOV     ST+1,R0                 ;GET TWO WORDS FROM PACKET
002507  104301  000746                  MOV     ST+2,R1
002511  102201  000250                  BIT     #ST.WE+ST.PE+ST.FE,R1   ;CHECK ERROR BITS
002513  012523                          BEQ     10$                     ;BRANCH IF ALL CLEAR
002514  117400  000742                  DEC     SAVSTA                  ;TRY ONCE MORE?
002516  052503                          BNE     DRCLR1                  ;IF OK, TRY AGAIN
002517  024265                          CALL    STOSTA                  ;GO STORE STATUS
002520                                  ERRHRD  MS2007
002520  025016                                                          CALL RERROR     ;ERROR # 2007.
002521  103727                                                          .WORD ERHARD+ERRN
002522  000421                                                          .WORD <PRMS*10000>+MS2007
                                ;
002523  104070  000742          10$:    MOV     R0,SAVSTA               ;SAVE STATUS TO CHECK IF DIAGNOSTIC REQUEST BIT IS S
                                ;
                                ;ISSUE ONLINE COMMAND AND CHECK THAT IT PERFORMS PROPERLY
                                ;
002525  104203  001445                  MOV     #CR.ONL,R3              ;R3 -> COMMAND
002527  024706                          CALL    TALKER                  ;SEND TO DRIVE
002530  115003                          TST     R3                      ;ALL OK?
002531  012546                          BEQ     30$                     ;IF SO, CHECK RESPONCE CODE
002532  032537                          BPL     20$                     ;ELSE, IS MSB SET?/IF NOT, RECEIVE ERROR
002533                                  ERRHRD  MS2008                  ;REPORT SEND ERROR
002533  025016                                                          CALL RERROR     ;ERROR # 2008.
002534  103730                                                          .WORD ERHARD+ERRN
002535  000500                                                          .WORD <PRMS*10000>+MS2008
002536  002574                          BR      GSTS7                   ;
002537  024504                  20$:    CALL    TYPERR
002540                                  ERRHRD  MS2009,R0,R3            ;REPORT RECEIVE ERROR
002540  100463                                                          MOV R3,-(SP)
002541  100467                                                          MOV R0,-(SP)
002542  025016                                                          CALL RERROR     ;ERROR # 2009.
002543  103731                                                          .WORD ERHARD+ERRN
002544  020530                                                          .WORD <PRMS*10000>+MS2009
002545  002574                          BR      GSTS7                   ;
002546  106207  000176          30$:    CMP     #COMPLT,R0              ;CHECK RESPONSE CODE
002550  012574                          BEQ     GSTS7                   ;IF OK, CONTINUE
002551  106207  000175                  CMP     #UNSSUC,R0              ;IF NOT CORRECT RESPONSE, UNSSUC?
002553  052561                          BNE     35$                     ;IF NOT, UNRECOGNIZED RESPONSE
```

```
     47 002554  024265                          CALL    STOSTA              ;CHECK STATUS
     48 002555                                  ERRHRD  MS2010              ;
        002555  025016                                                              CALL RERROR    ;ERROR # 2010.
        002556  103732                                                              .WORD ERHARD+ERRN
        002557  000564                                                              .WORD <PRMS*10000>+MS2010
     49 002560  002574                          BR      GSTS7
     50 002561                          35$:    ERRHRD  MS2011,#COMPLT,R0   ;IF NOT, REPORT ERROR
        002561  100467                                                              MOV R0,-(SP)
        002562  104010  001153                                                      MOV R1,SAVREG
        002564  104201  000176                                                      MOV #COMPLT,R1
        002566  100461                                                              MOV R1,-(SP)
        002567  104301  001153                                                      MOV SAVREG,R1
        002571  025016                                                              CALL RERROR    ;ERROR # 2011.
        002572  103733                                                              .WORD ERHARD+ERRN
        002573  020607                                                              .WORD <PRMS*10000>+MS2011
     51
     52 002574                          GSTS7:
```

```
                                        .SBTTL  GET DRIVE CHARACTERISTICS
                                ;GET DRIVE CHARACTERISTICS

002574  104200  000012  001154        MOV     #10.,SDISTO             ; SET UP TEMPORARY SHORT TIMEOUT VALUE
002577  104203  001437                MOV     #CR.GCR,R3              ;POINT TO GET CHARS COMMAND
002601  024706                        CALL    TALKER                  ; INITIATE SDI INTERCHANGE
002602  115003                        TST     R3                      ;SEE IF ERROR OCCURRED
002603  012620                        BEQ     12$                     ;IF NOT, BRANCH
002604  032611                        BPL     11$                     ;IF SO, BRANCH
002605                                ERRHRD  MS2012                  ;SEND COMMAND ERROR
002605  025016                                                                  CALL RERROR     ;ERROR # 2012.
002606  103734                                                                  .WORD ERHARD+ERRN
002607  000674                                                                  .WORD <PRMS*10000>+MS2012
002610  002646                        BR      TOO
002611  024504                11$:    CALL    TYPERR                  ;CHECK WHAT TYPE OF RECEIVE  RROR
002612                                ERRHRD  MS2013,R0,R3
002612  100463                                                                  MOV R3,-(SP)
002613  100467                                                                  MOV R0,-(SP)
002614  025016                                                                  CALL RERROR     ;ERROR # 2013.
002615  103735                                                                  .WORD ERHARD+ERRN
002616  020735                                                                  .WORD <PRMS*10000>+MS2013
002617  002646                        BR      TOO
002620  106207  000170        12$:    CMP     #CHRRES,R0              ;CHECK FOR SUCCESSFUL RESPONSE
002622  012646                        BEQ     TOO
002623  106207  000175                CMP     #UNSSUC,R0              ;IF NOT CORRECT RESPONSE, UNSSUC?
002625  052633                        BNE     13$                     ;IF NOT, UNRECOGNIZED RESPONSE
002626  024265                        CALL    STOSTA                  ;CHECK STATUS
002627                                ERRHRD  MS2014                  ;
002627  025016                                                                  CALL RERROR     ;ERROR # 2014.
002630  103736                                                                  .WORD ERHARD+ERRN
002631  001001                                                                  .WORD <PRMS*10000>+MS2014
002632  002646                        BR      TOO
002633                        13$:    ERRHRD  MS2015,#CHRRES,R0       ;GET CHARACTERISTICS COMMAND FAILED
002633  100467                                                                  MOV R0,-(SP)
002634  104010  001153                                                          MOV R1,SAVREG
002636  104201  000170                                                          MOV #CHRRES,R1
002640  100461                                                                  MOV R1,-(SP)
002641  104301  001153                                                          MOV SAVREG,R1
002643  025016                                                                  CALL RERROR     ;ERROR # 2015.
002644  103737                                                                  .WORD ERHARD+ERRN
002645  021035                                                                  .WORD <PRMS*10000>+MS2015

002646  104307  000744        TOO:    MOV     ST+SHRTTO,R0            ; GET SHORT TIMEOUT
002650  103207  177760                BIC     #LBLONB,R0              ; CLEAR UNUSED BITS
002652  025001                        CALL    TO                      ; SET UP TIMEOUT
002653  104070  001154                MOV     R0,SDISTO               ; SAVE IN SHORT TIMEOUT
002655  104307  000745                MOV     ST+LONGTO,R0            ; GET LONG TIMEOUT
002657  103207  177760                BIC     #LBLONB,R0              ; CLEAR UNUSED BITS
002661  025001                        CALL    TO                      ; SET UP TIMEOUT
002662  104070  001155                MOV     R0,SDILTO               ; SAVE IN LONG TIMEOUT
```

```
                                                .SBTTL  CHECK WHICH COMMAND HAS BEEN GIVEN
002664  104307  000742          ST.DIA: MOV     SAVSTA,R0               ;GET STORED STATUS WORD
002666  102207  000040                  BIT     #ST.DR,R0               ;WAS THE DR BIT SET?
002670  053306                          BNE     DO.DIX                  ;IF SO, GO HANDLE IT WITHIN DIAGNOSE CODE
                                        ;; *** DO AN INITIAL WRITE MEMORY TO CLEAR THE REGION (ONLY 4 BYTES)
002671  104200  177774  001213          MOV     #FFFC,WRM+1             ;REGION ID
002674  114000  001214                  CLR     WRM+2                   ;OFFSET
002676  104200  000004  001215          MOV     #4,WRM+3                ;BYTE COUNT AND 1ST DATA BYTE
002701  114000  001216                  CLR     WRM+4                   ;DATA
002703  114000  001217                  CLR     WRM+5                   ;DATA
002705  104200  000012  001206          MOV     #10.,CR.WRM+1           ;SET COMMAND BYTE COUNT
002710  023721                          CALL    WRTMEM                  ;CLEAR BUFFER
002711  104200  000007  001206          MOV     #7,CR.WRM+1             ;SET COMMAND BYTE COUNT
                                        ; *** NOW GO DIAGNOSE REGION ZERO
002714  114000  001501                  CLR     OUT.01                  ;CLEAR OUT BUFFER TO INITIAL DIAGNOSE COMMANDS
002716  114000  001502                  CLR     OUT.02                  ;
002720  114000  001144                  CLR     DIAG.1                  ;MAKE SURE DIAGNOSE COMMAND RETURNS OP-CODE = 0
                                                                        ; (FOR 1ST TIME THROUGH ONLY)
002722  114000  001545                  CLR     IN.02                   ;CLEAR REGION ID FOR 1ST DIAGNOSE COMMAND
002724  003054                          BR      DIAGNS                  ;DO DIAGNOSE COMMAND, 1ST THING
                                        ;
                                        ; *** RETURN HERE TO GET NEW COMMAND
002725                          GT.CMD:                                 ;
002725  104207  060002                  MOV     #T2CMD,R0               ;SET REQUEST NUMBER
002727  024643                          CALL    HCSTRQ                  ;ASK HOST FOR PORTS
002730  104307  001544                  MOV     IN.01,R0                ;R0 = RESPONSE CODE;IN.02
                                                                        ; 0 = EXIT ; 1 = WRITE
                                                                        ; 2 = READ ; 3 = DIAGNOSE
002732  013100                          BEQ     TESTEX                  ; OP = 0 /EXIT
002733  117407                          DEC     R0                      ; R0 = 1?
002734  012754                          BEQ     WRITE                   ; IF SO, WRITE
002735  117407                          DEC     R0                      ; R0 = 2?
002736  013020                          BEQ     READ                    ; IF SO, READ
002737  117407                          DEC     R0                      ; R0 = 3?
002740  013054                          BEQ     DIAGNS                  ; IF SO, DIAGNOSE
002741                                  ERRHRD  MS2016,IN.01            ; ELSE, ERROR=INVALID INPUT
002741  104010  001153                                                                  MOV R1,SAVREG
002743  104301  001544                                                                  MOV IN.01,R1
002745  100461                                                                          MOV R1,-(SP)
002746  104301  001153                                                                  MOV SAVREG,R1
002750  025016                                                                          CALL RERROR     :ERROR # 2016.
002751  103740                                                                          .WORD ERHARD+ERRN
002752  011133                                                                          .WORD <PRMS*10000>+MS2016
002753  002725                          BR      GT.CMD
```

```
                                                .SBTTL  MEMORY WRITE
                                        ;:+
                                        ;
                                        ;       INPUT   IN.02 = REGION ID
                                        ;               IN.03 = OFFSET
                                        ;               IN.04 = DATA BYTE
                                        ;
                                        ;       OUTPUT  OUT.RQ HAS DAT SET FOR T2CMD
                                        ;:-
002754                                  WRITE:
002754  104300  001545  001213                MOV     IN.02,WRM+1             ;REGION ID
002757  104300  001546  001214                MOV     IN.03,WRM+2             ;OFFSET
002762  104307  001547                        MOV     IN.04,R0                ;R0 = BYTE OF DATA IN LO BYTE
002764  110707                                SWAB    R0                      ;R0 IS IN HI BYTE
002765  103207  000377                        BIC     #LOBYTE,R0              ;CLEAR LO BYTE
002767  101207  000001                        BIS     #1,R0                   ;SET BYTE COUNT
002771  104070  001215                        MOV     R0,WRM+3                ;SET WORD IN PACKET
002773  114000  001216                        CLR     WRM+4
002775  114000  001217                        CLR     WRM+5
002777  104200  000007  001206                MOV     #7,CR.WRM+1             ;SET COMMAND BYTE COUNT
                                        ;
003002  023721                          CALL    WRTMEM
003003  104200  000001  001502                MOV     #1,OUT.02               ;R0 = OP CODE
003006  104300  001152  001501                MOV     LUNIT,OUT.01            ;DRIVE NUMBER
003011  114000  001503                        CLR     OUT.03                  ;
003013  114000  001504                        CLR     OUT.04                  ;
003015  114000  001505                        CLR     OUT.05                  ;
003017  002725                          BR      GT.CMD                  ;GO SEND REQUEST
```

```
                                                        .SBTTL  MEMORY READ
                                                ;;+
                                                ;
                                                ;       INPUT   IN.02 HAS REGION ID
                                                ;               IN.03 HAS OFFSET
                                                ;
                                                ;       OFFSET  SETS UP OUTBUFFER FOR T2CMD
                                                ;;-
 003020                                         READ:
 003020  104300  001545  001427                         MOV     IN.02,RDM+1             ;REGION ID
 003023  104300  001546  001430                         MOV     IN.03,RDM+2             ;OFFSET
 003026  104200  000001  001431                         MOV     #1,RDM+3                ;BYTE COUNT
                                                ;
 003031  023633                                         CALL    RDMEM
                                                ; *** SET UP RESPONSE PACKET
 003032  104300  001152  001501                         MOV     LUNIT,OUT.01            ;SET UP REQUEST
 003035  104200  000002  001502                         MOV     #2,OUT.02               ;RETURN OPCODE SET IN BUFFER
 003040  104307  000744                                 MOV     ST,R0                   ;R0 = BYTE COUNT + DATA
 003042  110707                                         SWAB    R0                      ;DATA IN LO BYTE
 003043  103207  177400                                 BIC     #HIBYTE,R0              ;CLEAR BYTE COUNT (1 BYTE ONLY SENT)
 003045  104070  001503                                 MOV     R0,OUT.03               ;STORE IN BUFFER FOR HOST
 003047  114000  001504                                 CLR     OUT.04                  ;
 003051  114000  001505                                 CLR     OUT.05                  ;
 003053  002725                                         BR      GT.CMD                  ;GO SEND REQUEST
```

```
                                                .SBTTL  SEND DIAGNOSE COMMAND
003054  102200  000001  000742  DIAGNS: BIT     #ST.RU,SAVSTA           ;CHECK IF DRIVE IS SPINNABLE (RUN SWITCH IN?)
003057  013074                          BEQ     3$                      ;ONLY DO DIAGNOSE ONCE IF NOT SPINNABLE
                                ; *** DRIVE IS SPINNABLE
003060  104302  000741                  MOV     SDI,R2                  ;R2 HAS CURRENT SDI PORT INDEX
003062  102200  000020  000742          BIT     #ST.SR,SAVSTA           ;HAS THE DRIVE BEEN SPUN UP?
003065  053073                          BNE     2$                      ;IF NOT ALREADY SPUN UP, DO DIAGNOSE TWICE
                                ; *** DRIVE HAS NOT BEEN SPUN UP
003066  023101                  1$:     CALL    DIAGDR                  ;DO A DIAGNOSE COMMAND
003067  104302  000741                  MOV     SDI,R2                  ;RESTORE PORT INDEX
003071  024001                          CALL    RUNDR                   ;SPIN UP THE DRIVE
003072  003074                          BR      3$
                                ; *** DRIVE HAS BEEN SPUN UP
003073  024055                  2$:     CALL    RECAL                   ;ELSE DO A RECAL
003074  104302  000741          3$:     MOV     SDI,R2                  ;RESTORE PORT INDEX
003076  023101                          CALL    DIAGDR                  ;DO A DIAGNOSE COMMAND
003077  002725                          BR      GT.CMD                  ;GET COMMAND

                                ;END OF TESTING THIS DRIVE

003100  002356                  TESTEX: BR TESTX                        ;GO BACK TO DRIVE
```

```
                                        .SBTTL  TEST 2 SPECIFIC ROUTINES
                                        .SBTTL  DIAGNOSE COMMAND PROCESSING
                                ;;+
                                ;
                                ;       INPUT   IN.02 HAS REGION ID
                                ;
                                ;       OUTPUT  OUT.RQ IS SET UP FOR T2CMD
                                ;               IF ERROR, R2 = 1
                                ;                   ELSE, R2 = 0
                                ;;-
003101                          DIAGDR:
003101  104200  177777  001151        MOV     #177777,SENDHR          ;SET SENDHR NONZERO FOR DIAGNOSE COMMAND
003104  104300  001545  001462        MOV     IN.02,DIA+1             ;SET UP MEMORY REGION ID
003107  104203  001454                MOV     #CR.DIA,R3              ;R3->DIAGNOSE PACKET
003111  104302  000741                MOV     SDI,R2                  ;R2 = PORT
003113  024706                        CALL    TALKER                  ;SEND COMMAND AND RECEIVE RESPONSE FROM DRIVE
003114  114000  001151                CLR     SENDHR
003116  115003                        TST     R3                      ;SEE IF ERROR OCCURRED
003117  013135                        BEQ     2$                      ;IF NO ERRORS, BRANCH
003120  110203                        ROL     R3                      ;SHIFT HIGH BIT INTO CARRY BIT
003121  043126                        BCC     1$                      ;IF CARRY CLEAR (RECEIVE ERROR) BRANCH
                                ; *** REPORT TRANSMISSION ERROR
003122                                  ERRHRD  MS2017
003122  025016                                                          CALL RERROR     ;ERROR # 2017.
003123  103741                                                          .WORD ERHARD+ERRN
003124  001224                                                          .WORD <PRMS*10000>+MS2017
003125  003166                        BR      10$
                                ; *** REPORT RECEPTION ERROR
003126  024504                  1$:   CALL    TYPERR
003127                                ERRHRD  MS2018,R0,R3
003127  100463                                                          MOV R3,-(SP)
003130  100467                                                          MOV R0,-(SP)
003131  025016                                                          CALL RERROR     ;ERROR # 2018.
003132  103742                                                          .WORD ERHARD+ERRN
003133  021255                                                          .WORD <PRMS*10000>+MS2018
003134  003166                        BR      10$
                                ; *** CHECK RESPONSE OP-CODE FROM DRIVE
003135                          2$:
003135  106207  000374                CMP     #DIA.EN,R0              ;CHECK RESPONSE CODE
003137  013171                        BEQ     3$                      ;IF EQUAL, BRANCH
003140  106207  000176                CMP     #COMPLT,R0              ; WAS IT COMPLETED?
003142  013171                        BEQ     3$                      ; IF SO, BRANCH
003143  106207  000175                CMP     #UNSSUC,R0              ;IF NOT CORRECT RESPONSE, UNSSUC?
003145  053153                        BNE     6$                      ;IF NOT, UNRECOGNIZED RESPONSE
003146  024265                        CALL    STOSTA                  ;CHECK STATUS
003147                                ERRHRD  MS2019                  ;
003147  025016                                                          CALL RERROR     ;ERROR # 2019.
003150  103743                                                          .WORD ERHARD+ERRN
003151  001312                                                          .WORD <PRMS*10000>+MS2019
003152  003166                        BR      10$
003153                          6$:   ERRHRD  MS2020,#DIA.EN,R0
003153  100467                                                          MOV R0,-(SP)
003154  104010  001153                                                  MOV R1,SAVREG
003156  104201  000374                                                  MOV #DIA.EN,R1
003160  100461                                                          MOV R1,-(SP)
003161  104301  001153                                                  MOV SAVREG,R1
003163  025016                                                          CALL RERROR     ;ERROR # 2020.
```

```
        003164  103744                                          .WORD ERHARD+ERRN
        003165  021336                                          .WORD <PRMS*10000>+MS2020
     41 003166  104202  000001          10$:    MOV     #1,R2
     42 003170  000000                          RETURN
```

```
                                                .SBTTL  DIAGNOSE/READ MEMORY TO SEE IF ERROR OCCURRED
                                        ; *** DO A READ MEMORY SDI COMMAND
003171  114000  001151                  3$:     CLR     SENDHR
003173  104300  000744  001427                  MOV     ST,RDM+1                ;REGION ID SET
003176  114000  001430                          CLR     RDM+2                   ;CLEAR OFFSET
003200  104200  000010  001431                  MOV     #8.,RDM+3               ;BYTE COUNT = 8
003203  023633                                  CALL    RDMEM                   ;READ THE MEMORY
003204  115002                                  TST     R2                      ;ALL OK?
003205  053263                                  BNE     DO.DC4                  ;IF NOT, DO DRIVE CLEAR/TRY NEXT DRIVE
003206  104302  000741                          MOV     SDI,R2                  ;RESTORE PORT INDICATOR
003210  104204  001505                          MOV     #OUT.05,R4              ;R4 -> OUT BUFFER
003212  024316                                  CALL    CONMEM                  ;CONVERT MEMORY
                                        ; *** DATA NOW IN OUT.RQ
003213  104303  001506                          MOV     OUT.06,R3               ;R3 HAD ET & DA FLAGS
                                        ; *** CHECK IF DATA IS AVAILABLE FOR INFORMATION
003215  102203  040000                          BIT     #DATAVL,R3              ;IS DATA AVAILABLE?
003217  013234                                  BEQ     5$                      ;IF NOT, CONTINUE WITH THIS DRIVE
003220  104200  000010  001430                  MOV     #8.,RDM+2               ;PREVIOUS BYTE COUNT
003223  024356                                  CALL    GETBCN                  ;GET NEW BYTE COUNT
003224  023633                                  CALL    RDMEM                   ;READ MEMORY XFC
003225  115002                                  TST     R2                      ;ALL OK?
003226  053263                                  BNE     DO.DC4                  ;IF NOT, DO DRIVE CLEAR/ELSE, DROP DRIVE
003227  104302  000741                          MOV     SDI,R2                  ;RESTORE PORT INDICATOR
003231  104204  001511                          MOV     #OUT.09,R4              ;R4 -> OUT BUFFER
003233  024316                                  CALL    CONMEM                  ;CONVERT MEMORY
003234  102203  100000                  5$:     BIT     #ERRTYP,R3              ;WAS ET SET? WITH NO ERROR NUMBER???
003236  013243                                  BEQ     4$                      ;IF NOT, CONTINUE
                                        ; *** REPORT AN ERROR
003237                                          ERRHRD  MS2021
003237  025016                                          CALL RERROR     ;ERROR # 2021.
003240  103745                                          .WORD ERHARD+ERRN
003241  001424                                          .WORD <PRMS*10000>+MS2021
003242  003263                                  BR      DO.DC4                  ;GO DO DRIVE CLEAR
003243  102203  040000                  4$:     BIT     #DATAVL,R3              ;IS DATA AVAILABLE?
003245  013263                                  BEQ     DO.DC4                  ;IF NOT, DO DRIVE CLEAR
003246                                          MSSGE   MSG1
003246  104300  001152  001501                          MOV     LUNIT,OUT.01
003251  104200  003242  001502                          MOV     #MSG1,OUT.02
003254  100467                                                  MOV R0,-(SP)
003255  100461                                                  MOV R1,-(SP)
003256  104207  060015                          MOV     #MESSAG,R0
003260  024643                                  CALL    HOSTRQ
003261  104261                                                  MOV (SP)+,R1
003262  104267                                                  MOV (SP)+,R0
```

```
                                                .SBTTL  DIAGNOSE/DO A DRIVE CLEAR
                                        ; *** DO DRIVE CLEAR, GET STATUS AND CHECK IF DR BIT IS SET
003263  104204  000004                  DO.DC4: MOV     #4,R4                   ;R4 = # MAXIMUM DRIVE CLEAR TRIES
003265  104302  000741                  DO.DCL: MOV     SDI,R2                  ;RESTORE PORT INDICATOR
003267  024211                                  CALL    CLRDRV                  ;CALL DRIVE CLEAR
003270  115003                                  TST     R3                      ;ERROR?
003271  013274                                  BEQ     1$                      ;IF NOT, CONTINUE
003272  117404                                  DEC     R4                      ;REACHED MAXIMUM # OF DRV CLR TRIES?
003273  053265                                  BNE     DO.DCL                  ;IF NOT, TRY AGAIN
003274  024131                          1$:     CALL    GTSTAT                  ;CALL GET STATUS
003275  115003                                  TST     R3                      ;ERROR?
003276  013301                                  BEQ     2$                      ;IF NOT, CONTINUE
003277  117404                                  DEC     R4                      ;REACHED MAXIMUM # OF DRV CLR TRIES?
003300  053265                                  BNE     DO.DCL                  ;IF NOT, TRY AGAIN
003301  104307  000745                  2$:     MOV     ST+1,R0                 ;GET STATUS WORD
003303  102207  000040                          BIT     #ST.DR,R0               ;DR SET?
003305  013603                                  BEQ     DO.DIO                  ;IF NOT, EXIT
                                        ; *** IF DR BIT IS SET, READ 6 BYTES FROM REGION ID FFFD
003306  104200  177775  001427          DO.DIX: MOV     #FFFD,RDM+1             ;FFFD (REGION ID) STORED IN READ MEM PACKET
003311  114000  001430                          CLR     RDM+2                   ;OFFSET = 0
003313  104200  000006  001431                  MOV     #6,RDM+3                ;BYTE COUNT = 6
003316  023633                                  CALL    RDMEM                   ;READ MEMORY
003317  115002                                  TST     R2                      ;WAS THERE AN ERROR?
003320  013322                                  BEQ     3$                      ;IF THERE WAS NOT, CONTINUE
003321  003603                                  BR      DO.DIO                  ;DO A DRIVE CLEAR
003322  104302  000741                  3$:     MOV     SDI,R2                  ;RESTORE PORT INDICATOR
003324  104204  000744                          MOV     #ST,R4                  ;R4 -> ST
003326  024316                                  CALL    CONMEM                  ;CONVERT MEMORY
                                        ; *** GET REGION ID AND PROGRAM NAME IF ANY
003327  104300  000744  000743                  MOV     ST,SAVRID               ;SAVE REGION ID
003332  115000  000745                          TST     ST+1                    ;IS CHARACTER ENCODED?
003334  053340                                  BNE     DO.DI1                  ;IF SO, GO GET FILE FROM HOST
003335  115000  000746                          TST     ST+2                    ;IF 1ST WORD 0, IS THE SECOND?
003337  013521                                  BEQ     DO.DI3                  ;IF SO, GO DIAGNOSE REGION
```

```
                                                .SBTTL  DIAGNOSE/GET PROGRAM NAME SPECIFIED BY DRIVE AND DOWNLINE LOAD
                                        ; *** GET THE PROGRAM WHICH NAME WAS SPECIFIED BY THE DRIVE
003340                                  DO.DI1:
003340  104300  000743  001213                MOV     SAVRID,WRM+1            ;STORE REGION ID IN THE WRITE PACKET
003343  104203  001501                        MOV     #OUT.01,R3              ;R3 -> INPUT DATA
003345  104307  001152                        MOV     LUNIT,R0
003347  100237                              1$: MOV     R0,(R3)+                ;SET UNIT NUMBER
003350  114007                                CLR     R0
003351  100237                                MOV     R0,(R3)+                ;CLEAR VALUE
003352  104307  000744                        MOV     ST,R0
003354  100237                                MOV     R0,(R3)+                ;SAVE REGION ID
003355  104307  000745                        MOV     ST+1,R0
003357  100237                                MOV     R0,(R3)+                ;1ST HALF NAME SAVED
003360  104307  000746                        MOV     ST+2,R0
003362  100237                                MOV     R0,(R3)+                ;2ND HALF NAME SAVED
003363  104207  060001                        MOV     #T2DLL,R0               ;SET R0 = TEST 2 DOWNLINE LOAD
003365  024643                                CALL    HOSTRQ
                                        ; *** IF HERE, DIDN'T REINIT UDA TO GET DATA
003366  104205  001544                        MOV     #IN.01,R5               ;R5 -> INPUT BUFFER
                                        ; *** DOWN LINE LOAD PROGRAM
003370  114000  001214                  DO.DI2: CLR     WRM+2                   ;CLEAR OFFSET
003372  104157                                MOV     (R5),R0                 ;IS THE PROGRAM THERE?
003373  117407                                DEC     R0
003374  053530                                BNE     DO.DI8                  ;IF NOT, REPORT ERROR
                                        ; *** GET THE PROGRAM FROM THE HOST
003375  104657  000004                  1$:   MOV     4(R5),R0                ;UNIBUS ADDRESS PA
003377  104651  000005                        MOV     5(R5),R1                ;UNIBUS ADDRESS EA
003401  104652  000006                        MOV     6(R5),R2                ;BYTE COUNT
003403  053407                                BNE     2$                      ;IF BYTE COUNT IS NOT ZERO, CONTINUE
003404                                        ERRHRD  MS2022                  ;ELSE, REPORT THE ERROR
003404  025016                                                                          CALL RERROR     ;ERROR # 2022.
003405  103746                                                                          .WORD ERHARD+ERRN
003406  001511                                                                          .WORD <PRMS*10000>+MS2022
003407  104203  000744                  2$:   MOV     #ST,R3                  ;POINTER TO INPUT BUFFER
003411  106202  000200                        CMP     #STSIZE,R2              ;R2 > MAX BUFFER SIZE?
003413  033416                                BPL     3$                      ;IF NOT, CONTINUE
003414  104202  000200                        MOV     #STSIZE,R2              ;SET MAX BUFFER SIZE IN R2
003416  060013                          3$:   XFC     UREAD                   ;DO A UNIBUS READ
                                        ; *** SET UP SDI COMMAND PACKETS
003417  104020  001206                        MOV     R2,CR.WRM+1             ;SET BYTE COUNT IN WRM PACKET
003421  105200  000006  001206                ADD     #6,CR.WRM+1             ;ADD TO COUNT TO INCLUDE PACKET LENGTH(1),
                                                                                ; SDI OPCODE(1), REGION ID(2), AND OFFSET(2)
003424  104650  000003  001213                MOV     3(R5),WRM+1             ;SET REGION ID
003427  104020  001215                        MOV     R2,WRM+3                ;SET BYTE COUNT IN BUFFER
003431  104303  000744                        MOV     ST,R3                   ;1ST DATA WORD IN R3
003433  110703                                SWAB    R3                      ;1ST DATA WORD IN UPPER BYTE
003434  103203  000377                        BIC     #LOBYTE,R3              ;CLEAR LOW BYTE
003436  101030  001215                        BIS     R3,WRM+3                ;SET DATA IN 1ST BUFFER WORD OF WRT MEM PACKET
003440  104020  001153                        MOV     R2,SAVREG               ;SAVE BYTE COUNT
003442  117402                                DEC     R2                      ;ADJUST BYTE COUNT
003443  103200  000377  000744                BIC     #LOBYTE,ST              ;CLEAR LOW BYTE OF 1ST DATA WORD OF PROGRAM
003446  101020  000744                        BIS     R2,ST                   ;REPLACE IT BY THE BYTE COUNT OF THE REST
003450  104204  001216                        MOV     #WRM+4,R4               ;R4 -> OUT BUFFER
003452  024316                                CALL    CONMEM                  ;CONVERT MEMORY
                                        ; *** SEND TO THE DRIVE
003453  104302  000741                        MOV     SDI,R2                  ;RESTORE PORT INDICATOR
003455  023721                                CALL    WRTMEM                  ;SEND TO DRIVE
```

```
     55 003456  115002                          TST     R2                      ;ALL OK?
     56 003457  053263                          BNE     DO.DC4                  ;IF NOT, GO CLEAR DRIVE
     57 003460  104302  001153                  MOV     SAVREG,R2               ;R2 = BYTE COUNT
     58 003462  105020  001214                  ADD     R2,WRM+2                ;ADJUST OFFSET
     59 003464  104657  000006                  MOV     6(R5),R0
     60 003466  107027                          SUB     R2,R0
     61 003467  100657  000006                  MOV     R0,6(R5)                ;DECREMENT TOTAL BYTE COUNT/DONE?
     62 003471  013510                          BEQ     5$                      ;IF 0, EXIT
     63 003472  073510                          BMI     5$                      ;IF NEG, EXIT
     64 003473  104657  000004                  MOV     4(R5),R0
     65 003475  105027                          ADD     R2,R0
     66 003476  100657  000004                  MOV     R0,4(R5)                ;ELSE, ADJUST UNIBUS ADDRESS TO READ FROM
     67 003500  106027                          CMP     R2,R0                   ;IS PA ADDRESS LESS THAN BUFFER SIZE?
     68 003501  073375                          BMI     1$                      ;IF IT IS NOT, CONTINUE
     69 003502  104657  000005                  MOV     5(R5),R0
     70 003504  115407                          INC     R0                      ;ELSE, INCREMENT EA ADDRESS (CROSSED 0 BOUNDARY)
     71 003505  100657  000005                  MOV     R0,5(R5)
     72 003507  003375                  4$:     BR      1$                      ;READ IN NEXT BUFFER
     73
     74                                 ; *** SET UP DIAGNOSE COMMAND
     75 003510  114000  001214          5$:     CLR     WRM+2                   ;REINIT OFFSET
     76 003512  104200  000007  001206          MOV     #7,CR.WRM+1             ;REINIT BYTE COUNT OF INSTRUCTION
     77 003515  104650  000003  001545          MOV     3(R5),IN.02             ;SET UP REGION ID
     78 003520  003101                          BR      DIAGDR                  ;SEND DIAGNOSE COMMAND
     79
     80                                 ; *** JUST RECEIVED THE REGION ID TO DIAGNOSE TO.  GO DIAGNOSE IT.
     81 003521  104300  000743  001545  DO.D13: MOV     SAVRID,IN.02            ;STORE REGION ID FOR DIAGNOSE COMMAND
     82 003524  003101                          BR      DIAGDR                  ;GO DIAGNOSE
```

```
                                        .SBTTL  DIAGNOSE/REPORT ERROR -- NO DOWNLINE LOAD PROGRAM
                                ; *** NO DOWN LINE LOAD PROGRAM AFTER REINITED UDA
003525  104650  000001  000740  DO.DI7: MOV     1(R5),UNITNB            ;SET UP LOGICAL UNIT NUMBER

                                ; *** NO DOWN LINE LOAD PROGRAM, REPORT ERROR
003530  104654  000007          DO.DI8: MOV     7(R5),R4                ;R4 = 1ST WORD OF PROGRAM NAME
003532  024544                          CALL    DIV50                   ;DIVIDE BY 50 AND GET 3RD CHARACTER
003533  110707                          SWAB    R0                      ;PUT IN PROPER BYTE
003534  104070  001146                  MOV     R0,NAM.2                ;SET 3RD CHARACTER
003536  024544                          CALL    DIV50                   ;GET 2ND CHARACTER
003537  104070  001147                  MOV     R0,NAM.3                ;SET 2ND CHARACTER
003541  104047                          MOV     R4,R0                   ;R0 = RAD50 OF 1ST CHARACTER
003542  024560                          CALL    FNDASC                  ;GET ASCII EQUIVALENT
003543  110707                          SWAB    R0                      ;PUT IN PROPER BYTE
003544  101070  001147                  BIS     R0,NAM.3                ;SET 1ST CHARACTER
003546  104654  000010                  MOV     10(R5),R4               ;R4 = 2ND WORD OF PROGRAM NAME
003550  024544                          CALL    DIV50                   ;GET 6TH CHARACTER
003551  104070  001145                  MOV     R0,NAM.1                ;SET 6TH CHARACTER
003553  024544                          CALL    DIV50                   ;GET 5TH CHARACTER
003554  110707                          SWAB    R0                      ;PUT IN PROPER BYTE
003555  101070  001145                  BIS     R0,NAM.1                ;SET 5TH CHARACTER
003557  104047                          MOV     R4,R0                   ;R0 = RAD50 OF 4TH CHARACTER
003560  024560                          CALL    FNDASC                  ;FIND ASCII EQUIVALENT
003561  101070  001146                  BIS     R0,NAM.2                ;SET 4TH CHARACTER
003563                                  ERRHRD  MS2023,NAM.1,NAM.2,NAM.3 ;REPORT ERROR
003563  104010  001153                                                          MOV R1,SAVREG
003565  104301  001147                                                          MOV NAM.3,R1
003567  100461                                                                  MOV R1,-(SP)
003570  104301  001146                                                          MOV NAM.2,R1
003572  100461                                                                  MOV R1,-(SP)
003573  104301  001145                                                          MOV NAM.1,R1
003575  100461                                                                  MOV R1,-(SP)
003576  104301  001153                                                          MOV SAVREG,R1
003600  025016                                                                  CALL RERROR     ;ERROR # 2023.
003601  103747                                                                  .WORD ERHARD+ERRN
003602  031554                                                                  .WORD <PRMS*10000>+MS2023
```

```
                                                .SBTTL  DIAGNOSE/SET UP RESPONSE TO HOST AND EXIT
                                        ; *** SET UP RESPONSE PACKET
003603  104302  000741          DO.DIO: MOV     SDI,R2                  ;RESTORE PORT INDICATOR
003605  104300  001152  001501          MOV     LUNIT,OUT.01            ;SAVE UNIT NUMBER
003610  115000  001144                  TST     DIAG.1                  ;IF DIAG.1 = 0?
003612  053616                          BNE     1$                      ;IF NOT, SKIP
003613  114000  001502                  CLR     OUT.02                  ;SET UP PROPER OP-CODE VALUE
003615  003621                          BR      2$                      ;CONTINUE
003616  104200  000003  001502  1$:     MOV     #3,OUT.02               ;SET UP REQUEST
003621                          2$:
003621  104300  000744  001503          MOV     ST,OUT.03               ;REGION ID SET
003624  114000  001504                  CLR     OUT.04                  ;
003626  104200  177777  001144          MOV     #177777,DIAG.1          ;MAKE SURE DIAG.1 HAS NONE ZERO VALUE IN IT
                                                                        ; FOR NEXT TIME THROUGH
003631  114002                          CLR     R2
003632  000000                          RETURN
```

```
                                        .SBTTL  READ MEMORY SUBROUTINE
                                ;:+
                                ;       ROUTINE NAME:   RDMEM
                                ;
                                ;       DESCRIPTION:
                                ;       THIS ROUTINE DOES THE XFC READ MEMORY (FROM A SPECIFIED REGION
                                ;       AND OFFSET) OF DATA FROM THE DRIVE.  THIS ROUTINE CALLS THE
                                ;       SEND AND RCV (TALKER) ROUTINE AND JUDGES IF THE SDI COMMAND
                                ;       WAS PROPERLY SENT.
                                ;
                                ;       INPUT:  ALL PARAMETERS FOR THE READ MEMORY COMMAND
                                ;               MUST BE SET.  THAT MEANS REGION ID, THE OFFSET,
                                ;               AND THE BYTE COUNT MUST BE GIVEN THE APPRORIATE VALUES
                                ;               AND STORED IN THE RDM PACKET
                                ;
                                ;       OUTPUT: R2 = 0 MEANS ALL IS OK
                                ;               R2 NOT = 0, READ FAILED (EITHER SEND OR RECIEVE OF THE COMMAND)
                                ;               ST HAS THE READ DATA
                                ;:-
003633                          RDMEM:
003633                                  PUSH    <R0,R1,R3>              ;SAVE THESE REGISTERS
003633  100467                                                                  MOV R0,-(SP)
003634  100461                                                                  MOV R1,-(SP)
003635  100463                                                                  MOV R3,-(SP)
003636  104203  001421                 MOV     #CR.RDM,R3              ;R3 -> RDM PACKET
003640  104302  000741                 MOV     SDI,R2                  ;R2 = PORT
003642  024706                         CALL    TALKER                  ;SEND COMMAND AND RECEIVE RESPONSE
003643  115003                         TST     R3                      ;ERRORS?
003644  013662                         BEQ     2$                      ;IF NONE SO FAR, CONTINUE
003645  110203                         ROL     R3                      ;ERROP ON SEND?
003646  043653                         BCC     1$                      ;IF CARRY CLEAR (RECEIVE ERROR) BRANCH
                                ; *** REPORT TRANSMISSION ERROR
003647                                  ERRHRD  MS2024
003647  025016                                                                  CALL RERROR     ;ERROR # 2024.
003650  103750                                                                  .WORD ERHARD+ERRN
003651  001621                                                                  .WORD <PRMS*10000>+MS2024
003652  003713                         BR      4$                      ;ERROR EXIT
                                ; *** REPORT RECEPTION ERROR
003653  024504                 1$:     CALL    TYPERR                  ;GO CHECK TYPE OF ERROR
003654                                  ERRHRD  MS2025,R0,R3
003654  100463                                                                  MOV R3,-(SP)
003655  100467                                                                  MOV R0,-(SP)
003656  025016                                                                  CALL RERROR     ;ERROR # 2025.
003657  103751                                                                  .WORD ERHARD+ERRN
003660  021653                                                                  .WORD <PRMS*10000>+MS2025
003661  003713                         BR      4$                      ;ERROR EXIT
                                ; *** CHECK RESPONSE OP-CODE FROM DRIVE
003662  106207  000162         2$:     CMP     #RDM.EN,R0              ;CHECK RESPONSE
003664  013711                         BEQ     3$                      ;IF IT MATCHES, OK, EXIT
003665  106207  000175                 CMP     #UNSSUC,R0              ;IF NOT CORRECT RESPONSE, UNSSUC?
003667  053675                         BNE     6$                      ;IF NOT, UNRECOGNIZED RESPONSE
003670  024265                         CALL    STOSTA                  ;CHECK STATUS
003671                                  ERRHRD  MS2026                  ;
003671  025016                                                                  CALL RERROR     ;ERROR # 2026.
003672  103752                                                                  .WORD ERHARD+ERRN
003673  001712                                                                  .WORD <PRMS*10000>+MS2026
003674  003713                         BR      4$
```

```
     44 003675                          6$:     ERRHRD  MS2027,#RDM.EN,R0
        003675  100467                                                      MOV R0,-(SP)
        003676  104010  001153                                             MOV R1,SAVREG
        003700  104201  000162                                             MOV #RDM.EN,R1
        003702  100461                                                      MOV R1,-(SP)
        003703  104301  001153                                             MOV SAVREG,R1
        003705  025016                                                      CALL RERROR    ;ERROR # 2027.
        003706  103753                                                      .WORD ERHARD+ERRN
        003707  021737                                                      .WORD <PRMS*10000>+MS2027
     45 003710  003713                          BR      4$                  ;ERROR EXIT
     46 003711  114002                  3$:     CLR     R2                  ;ALL OK, EXIT
     47 003712  003715                          BR      5$                  ;
     48 003713  104202  000001          4$:     MOV     #1,R2               ;ERROR OCCURED, SET R2
     49 003715                          5$:     POP     <R3,R1,R0>
        003715  104263                                                              MOV (SP)+,R3
        003716  104261                                                              MOV (SP)+,R1
        003717  104267                                                              MOV (SP)+,R0
     50 003720  000000                          RETURN
```

```
                                        .SBTTL  WRITE MEMORY SUBROUTINE
                                ;;*
                                ;       WRTMEM
                                ;
                                ;       DESCRIPTION:
                                ;       THIS ROUTINE DOES THE XFC WRITE MEMORY TO A SPECIFIED
                                ;       REGION ID AND OFFSET WITH A BYTE COUNT AND DATA.
                                ;       THIS ROUTINE CALLS THE SEND AND RCV (TALKER) ROUTINE
                                ;       AND JUDGES IF THE SDI COMMAND WAS PROPERLY SENT.
                                ;
                                ;       INPUT:  ALL PARAMETERS FOR THE WRITE MEMORY COMMAND MUST BE SET.
                                ;               REGION ID, OFFSET, BYTE COUNT AND THE DATA MUST BE STORED
                                ;               IN THE PACKET.
                                ;
                                ;       OUTPUT: R2 = 0 MEANS WRITE MEMORY COMMAND WAS PROCESSED SUCCESSFULLY
                                ;               R2 NOT = 0, WRITE FAILED (EITHER SEND OR RECEIVE OF THE COMMAND)
                                ;;-
003721  104203  001205        WRTMEM: MOV     #CR.WRM,R3              ;R3->MEM WRITE PACKET
003723  104302  000741                MOV     SDI,R2                  ;R2 = PORT
003725  024706                       CALL    TALKER                  ;SEND COMMAND AND RECEIVE RESPONSE FROM DRIVE
003726  115003                       TST     R3                      ;SEE IF ERROR OCCURRED
003727  013745                       BEQ     2$                      ;IF NO ERRORS, BRANCH
003730  110203                       ROL     R3                      ;SHIFT HIGH BIT INTO CARRY BIT
003731  043736                       BCC     1$                      ;IF CARRY CLEAR (RECEIVE ERROR) BRANCH
                                ; *** REPORT TRANSMISSION ERROR
003732                               ERRHRD  MS2028
003732  025016                                                               CALL RERROR    ;ERROR # 2028.
003733  103754                                                               .WORD ERHARD+ERRN
003734  002027                                                               .WORD <PRMS*10000>+MS2028
003735  003776                       BR      4$
                                ; *** REPORT RECEPTION ERROR
003736  024504                  1$:  CALL    TYPERR                  ;FIND OUT WHAT TYPE OF ERROR OCCURED
003737                               ERRHRD  MS2029,R0,R3
003737  100463                                                               MOV R3,-(SP)
003740  100467                                                               MOV R0,-(SP)
003741  025016                                                               CALL RERROR    ;ERROR # 2029.
003742  103755                                                               .WORD ERHARD+ERRN
003743  022062                                                               .WORD <PRMS*10000>+MS2029
003744  003776                       BR      4$
                                ; *** SET UP RESPONSE PACKET
003745  106207  000176          2$:  CMP     #COMPLT,R0              ;DID IT COMPLETE
003747  013774                       BEQ     3$                      ;IF SO CONTINUE
003750  106207  000175                CMP     #UNSSUC,R0              ;IF NOT CORRECT RESPONSE, UNSSUC?
003752  053760                       BNE     6$                      ;IF NOT, UNRECOGNIZED RESPONSE
003753  024265                       CALL    STOSTA                  ;CHECK STATUS
003754                               ERRHRD  MS2030                  ;
003754  025016                                                               CALL RERROR    ;ERROR # 2030.
003755  103756                                                               .WORD ERHARD+ERRN
003756  002121                                                               .WORD <PRMS*10000>+MS2030
003757  003776                       BR      4$
003760                          6$:  ERRHRD  MS2031,#COMPLT,R0
003760  100467                                                               MOV R0,-(SP)
003761  104010  001153                                                       MOV R1,SAVREG
003763  104201  000176                                                       MOV #COMPLT,R1
003765  100461                                                               MOV R1,-(SP)
003766  104301  001153                                                       MOV SAVREG,R1
003770  025016                                                               CALL RERROR    ;ERROR # 2031.
```

```
                003771  103757                                                  .WORD ERHARD+ERRN
                003772  022147                                                  .WORD <PRMS*10000>+MS2031
     41         003773  003776                          BR      4$              ;ERROR EXIT
     42         003774  114002                  3$:     CLR     R2              ;ALL OK, EXIT
     43         003775  004000                          BR      5$              ;
     44         003776  104202  000001          4$:     MOV     #1,R2           ;ERROR OCCURED, SET R2
     45         004000  000000                  5$:     RETURN
```

RUN (OR SPIN UP) COMMAND

```
                                        .SBTTL  RUN (OR SPIN UP) COMMAND
                                ;ISSUE RUN COMMAND AND CHECK THAT IT PERFORMS PROPERLY
                                
004001  104203  001464  RUNDR: MOV     #CR.RUN,R3              ;R3 -> COMMAND
004003  024706                CALL    TALKER                  ;SEND TO DRIVE
004004  115003                TST     R3                      ;ALL OK?
004005  014022                BEQ     60$                     ;IF SO, CHECK RESPONCE CODE
004006  034013                BPL     50$                     ;ELSE, IS MSB SET?/IF NOT, RECEIVE ERROR
004007                        ERRHRD  MS2032                  ;REPORT SEND ERROR
004007  025016                                                                CALL RERROR     ;ERROR # 2032.
004010  103760                                                                .WORD ERHARD+ERRN
004011  002237                                                                .WORD <PRMS*10000>+MS2032
004012  004050                BR      70$                     ;
004013  024504          50$:  CALL    TYPERR
004014                        ERRHRD  MS2033,R0,R3            ;REPORT RECEIVE ERROR
004014  100463                                                                MOV R3,-(SP)
004015  100467                                                                MOV R0,-(SP)
004016  025016                                                                CALL RERROR     ;ERROR # 2033.
004017  103761                                                                .WORD ERHARD+ERRN
004020  022265                                                                .WORD <PRMS*10000>+MS2033
004021  004050                BR      70$                     ;
004022  106207  000176  60$:  CMP     #COMPLT,R0              ;CHECK RESPONSE CODE
004024  014053                BEQ     75$                     ;IF OK, CONTINUE
004025  106207  000175        CMP     #UNSSUC,R0              ;IF NOT CORRECT RESPONSE, UNSSUC?
004027  054035                BNE     65$                     ;IF NOT, UNRECOGNIZED RESPONSE
004030  024265                CALL    STOSTA                  ;CHECK STATUS
004031                        ERRHRD  MS2034                  ;
004031  025016                                                                CALL RERROR     ;ERROR # 2034.
004032  103762                                                                .WORD ERHARD+ERRN
004033  002317                                                                .WORD <PRMS*10000>+MS2034
004034  004050                BR      70$
004035                  65$:  ERRHRD  MS2035,#COMPLT,R0       ;IF NOT, REPORT ERROR
004035  100467                                                                MOV R0,-(SP)
004036  104010  001153                                                        MOV R1,SAVREG
004040  104201  000176                                                        MOV #COMPLT,R1
004042  100461                                                                MOV R1,-(SP)
004043  104301  001153                                                        MOV SAVREG,R1
004045  025016                                                                CALL RERROR     ;ERROR # 2035.
004046  103763                                                                .WORD ERHARD+ERRN
004047  022340                                                                .WORD <PRMS*10000>+MS2035
004050  104202  000001  70$:  MOV     #1,R2                   ;R2 IS NOT ZERO MEANING ERROR
004052  004054                BR      80$
004053  114002          75$:  CLR     R2                      ;R2 IS ZERO, NOT ERROR
004054  000000          80$:  RETURN
```

```
                                        .SBTTL  RECALIBRATE COMMAND
                                ;ISSUE RECALIBRATE COMMAND AND CHECK THAT IT PERFORMS PROPERLY
                                ;
004055  104203  001472    RECAL:  MOV     #CR.INR,R3              ;R3 -> COMMAND
004057  024706                    CALL    TALKER                  ;SEND TO DRIVE
004060  115003                    TST     R3                      ;ALL OK?
004061  014076                    BEQ     90$                     ;IF SO, CHECK RESPONCE CODE
004062  034067                    BPL     80$                     ;ELSE, IS MSB SET?/IF NOT, RECEIVE ERROR
004063                            ERRHRD  MS2036                  ;REPORT SEND ERROR
004063  025016                                                                    CALL RERROR     ;ERROR # 2036.
004064  103764                                                                    .WORD ERHARD+ERRN
004065  002424                                                                    .WORD <PRMS*10000>+MS2036
004066  004124                    BR      100$                    ;
004067  024504            80$:    CALL    TYPERR
004070                            ERRHRD  MS2037,R0,R3            ;REPORT RECEIVE ERROR
004070  100463                                                                    MOV R3,-(SP)
004071  100467                                                                    MOV R0,-(SP)
004072  025016                                                                    CALL RERROR     ;ERROR # 2037.
004073  103765                                                                    .WORD ERHARD+ERRN
004074  022456                                                                    .WORD <PRMS*10000>+MS2037
004075  004124                    BR      100$                    ;
004076  106207  000176    90$:    CMP     #COMPLT,R0              ;CHECK RESPONSE CODE
004100  014127                    BEQ     105$                    ;IF OK, CONTINUE
004101  106207  000175            CMP     #UNSSUC,R0              ;IF NOT CORRECT RESPONSE, UNSSUC?
004103  054111                    BNE     95$                     ;IF NOT, UNRECOGNIZED RESPONSE
004104  024265                    CALL    STOSTA                  ;CHECK STATUS
004105                            ERRHRD  MS2038                  ;
004105  025016                                                                    CALL RERROR     ;ERROR # 2038.
004106  103766                                                                    .WORD ERHARD+ERRN
004107  002514                                                                    .WORD <PRMS*10000>+MS2038
004110  004124                    BR      100$
004111                    95$:    ERRHRD  MS2039,#COMPLT,R0       ;IF NOT, REPORT ERROR
004111  100467                                                                    MOV R0,-(SP)
004112  104010  001153                                                            MOV R1,SAVREG
004114  104201  000176                                                            MOV #COMPLT,R1
004116  100461                                                                    MOV R1,-(SP)
004117  104301  001153                                                            MOV SAVREG,R1
004121  025016                                                                    CALL RERROR     ;ERROR # 2039.
004122  103767                                                                    .WORD ERHARD+ERRN
004123  022541                                                                    .WORD <PRMS*10000>+MS2039
004124  104202  000001    100$:   MOV     #1,R2
004126  004130                    BR      110$
004127  114002            105$:   CLR     R2
004130  000000            110$:   RETURN
```

```
                                .SBTTL  GET STATUS SUBROUTINE
                        ;
                        ;       GET STATUS
                        ;
                        ;       OUTPUT  R3 = 0 IF OK
                        ;               R3 = 1 IF ERROR
                        ;               R4 USED AS A DECREMENT COUNTER
                        ;
004131  114004          GTSTAT: CLR     R4
004132  104203  001170  3$:     MOV     #CR.GST,R3              ;POINT TO GET STATUS COMMAND
004134  024706                 CALL    TALKER                  ; SEND AND RECEIVE COMMAND
004135  115003                 TST     R3                      ; SEE IF ERROR OCCURED
004136  014156                 BEQ     2$                      ; IF NOT, BRANCH
004137  110203                 ROL     R3                      ; SEE IF RECEIVE ERROR
004140  044147                 BCC     1$                      ; IF SO, BRANCH
004141  117404                 DEC     R4                      ; IF SEND ERROR, COUNTER DONE?
004142  054132                 BNE     3$                      ; IF NOT, TRY AGAIN
004143                         ERRHRD  MS2040
004143  025016                                                                  CALL RERROR     ;ERROR # 2040.
004144  103770                                                                  .WORD ERHARD+ERRN
004145  002631                                                                  .WORD <PRMS*10000>+MS2040
004146  004204                 BR      ER.END
004147  024504         1$:     CALL    TYPERR
004150                         ERRHRD  MS2041,R0,R3
004150  100463                                                                  MOV R3,-(SP)
004151  100467                                                                  MOV R0,-(SP)
004152  025016                                                                  CALL RERROR     ;ERROR # 2041.
004153  103771                                                                  .WORD ERHARD+ERRN
004154  022663                                                                  .WORD <PRMS*10000>+MS2041
004155  004204                 BR      ER.END
                        ; *** LOOK AT RESPONSE OP-CODE FROM DRIVE -- SHOULD BE DATA RESPONSE
004156  106207  000366  2$:     CMP     #STSRES,R0              ;CHECK OP-CODE
004160  014207                 BEQ     OK.END                  ;BRANCH IF A MATCH
004161  106207  000175         CMP     #UNSSUC,R0              ;IF NOT CORRECT RESPONSE, UNSSUC?
004163  054171                 BNE     4$                      ;IF NOT, UNRECOGNIZED RESPONSE
004164  024265                 CALL    STOSTA
004165                         ERRHRD  MS2042                  ;
004165  025016                                                                  CALL RERROR     ;ERROR # 2042.
004166  103772                                                                  .WORD ERHARD+ERRN
004167  002721                                                                  .WORD <PRMS*10000>+MS2042
004170  004204                 BR      ER.END
004171                 4$:     ERRHRD  MS2043,#STSRES,R0       ;WRONG RESPONSE FROM DRIVE
004171  100467                                                                  MOV R0,-(SP)
004172  104010  001153                                                          MOV R1,SAVREG
004174  104201  000366                                                          MOV #STSRES,R1
004176  100461                                                                  MOV R1,-(SP)
004177  104301  001153                                                          MOV SAVREG,R1
004201  025016                                                                  CALL RERROR     ;ERROR # 2043.
004202  103773                                                                  .WORD ERHARD+ERRN
004203  022746                                                                  .WORD <PRMS*10000>+MS2043
004204  104203  000001  ER.END: MOV     #1,R3                   ;R3 = NONE ZERO VALUE WHEN DONE(ERROR)
004206  004210                 BR      EX.END                  ;EXIT
004207  114003                 OK.END: CLR     R3              ;R3 = 0 WHEN DONE (NO ERROR)
004210  000000                 EX.END: RETURN
```

```
                                        .SBTTL  CLEAR DRIVE SUBROUTINE
                                ;
                                ;       DRIVE CLEAR
                                ;
                                ;       OUTPUT  R3 = 0 IF OK
                                ;               R3 = 1 IF ERROR
                                ;               R4 USED AS A DECREMENT COUNTER
                                ;
004211  114004          CLRDRV: CLR     R4                      ; CLEAR DECEMENT COUNTER
004212  104203  001176    3$:   MOV     #CR.DRC,R3              ; POINT TO DRIVE CLEAR COMMAND
004214  024706                  CALL    TALKER                  ; SEND AND RECEIVE COMMAND
004215  115003                  TST     R3                      ; SEE IF ERROR OCCURED
004216  014236                  BEQ     2$                      ; IF NOT, BRANCH
004217  110203                  ROL     R3                      ; SEE IF RECEIVE ERROR
004220  044227                  BCC     1$                      ; IF SO, BRANCH
004221  117404                  DEC     R4                      ; ON SEND ERROR, IS COUNTER DONE?
004222  054212                  BNE     3$                      ; IF NOT, TRY AGAIN
004223                          ERRHRD  MS2044
004223  025016                                                                  CALL RERROR     ;ERROR # 2044.
004224  103774                                                                  .WORD ERHARD+ERRN
004225  003035                                                                  .WORD <PRMS*10000>+MS2044
004226  004204                  BR      ER.END
004227  024504          1$:     CALL    TYPERR
004230                          ERRHRD  MS2045,R0,R3
004230  100463                                                                  MOV R3,-(SP)
004231  100467                                                                  MOV R0,-(SP)
004232  025016                                                                  CALL RERROR     ;ERROR # 2045.
004233  103775                                                                  .WORD ERHARD+ERRN
004234  023067                                                                  .WORD <PRMS*10000>+MS2045
004235  004204                  BR      ER.END
                                ; *** CHECK RESPONSE CODE
004236  106207  000176    2$:   CMP     #COMPLT,R0              ; SEE IF CORRECT RESPONSE CODE RECEIVED
004240  014207                  BEQ     OK.END                  ; IF SUCCESSFUL, BRANCH
004241  106207  000175          CMP     #UNSSUC,R0              ;IF NOT CORRECT RESPONSE, UNSSUC?
004243  054251                  BNE     4$                      ;IF NOT, UNRECOGNIZED RESPONSE
004244  024265                  CALL    STOSTA                  ;CHECK STATUS
004245                          ERRHRD  MS2046                  ;
004245  025016                                                                  CALL RERROR     ;ERROR # 2046.
004246  103776                                                                  .WORD ERHARD+ERRN
004247  003125                                                                  .WORD <PRMS*10000>+MS2046
004250  004204                  BR      ER.END
004251                  4$:     ERRHRD  MS2047,#COMPLT,R0       ;CLEAR ERROR FAILED
004251  100467                                                                  MOV R0,-(SP)
004252  104010  001153                                                          MOV R1,SAVREG
004254  104201  000176                                                          MOV #COMPLT,R1
004256  100461                                                                  MOV R1,-(SP)
004257  104301  001153                                                          MOV SAVREG,R1
004261  025016                                                                  CALL RERROR     ;ERROR # 2047.
004262  103777                                                                  .WORD ERHARD+ERRN
004263  023152                                                                  .WORD <PRMS*10000>+MS2047
004264  004204                  BR      ER.END
```

```
      1                                          .SBTTL  STORE STATUS ROUTINE
      2                                ;         STORE STATUS
      3                                ;               MOVE WHAT DATA IS IN ST AND PUT IT IN OUTPUT BUFFER
      4 004265                         STOSTA:
      5 004265                                 PUSH    <R0,R1,R2>
        004265  100467                                                                  MOV R0,-(SP)
        004266  100461                                                                  MOV R1,-(SP)
        004267  100462                                                                  MOV R2,-(SP)
      6 004270  104207  001505                   MOV     #OUT.05,R0           ;R0 -> OUTPUT BUFFER
      7 004272  104201  004252                   MOV     #STAT0,R1            ;MOVE FORMAT ADDRESS IN OUTPUT BUFFER
      8 004274  100271                           MOV     R1,(R0)+
      9 004275  104302  000741                   MOV     SDI,R2               ; R2 PORT INDICATOR
     10 004277  060007                           XFC     STATUS               ; GET STATUS
     11 004300  103201  077270                   BIC     #^C100507,R1         ; CLEAR UNUSED BITS
     12 004302  100271                           MOV     R1,(R0)+             ; STORE
     13 004303  104201  000753                   MOV     #ST+7,R1             ;R1 -> INPUT BUFFER
     14 004305  104412                  1$:      MOV     -(R1),R2             ;SAVE
     15 004306  100272                           MOV     R2,(R0)+
     16 004307  106201  000744                   CMP     #ST,R1               ;DONE?
     17 004311  054305                           BNE     1$                   ;IF NOT, CONTINUE
     18 004312                                   POP     <R2,R1,R0>
        004312  104262                                                                  MOV (SP)+,R2
        004313  104261                                                                  MOV (SP)+,R1
        004314  104267                                                                  MOV (SP)+,R0
     19 004315  000000                           RETURN
```

```
                                        .SBTTL  CONVERT MEMORY -- SKEWED BY BYTE
                                ;:+
                                ;       ROUTINE NAME:   CONMEM
                                ;
                                ;       DESCRIPTION:
                                ;       MEMORY IS SENT TO THE DM ROUTINE STARTING ON THE ODD BYTE BOUNDARY.
                                ;       THIS ROUTINE SHIFTS THE DATA TO START ON THE EVEN BYTE BOUNDARY AND
                                ;       STORES EACH WORD INTO A SPECIFIED BUFFER AREA.  THE DATA IS STORED AS
                                ;       FOLLOWS:
                                ;               ST:     DATA BYTE 0     BYTE COUNT
                                ;                       DATA BYTE 2     DATA BYTE 1
                                ;                       DATA BYTE 4     DATA BYTE 3
                                ;                       .....           DATA BYTE 5
                                ;
                                ;       AFTER EXECUTION, THD DATA IS STORED LIKE THIS:
                                ;               OUT.XX: DATA BYTE 1     DATA BYTE 0
                                ;                       DATA BYTE 3     DATA BYTE 2
                                ;                       DATA BYTE 5     DATA BYTE 4
                                ;                       .....           DATA BYTE 6
                                ;
                                ;       INPUT:  R4 -> BUFFER AREA'S FIRST LOCATION
                                ;               ST HAS INPUT DATA
                                ;
                                ;       OUTPUT: OUTPUT BUFFER HAS SHIFTED DATA
                                ;
                                ;:-
004316                          CONMEM: PUSH    <R0,R1,R2,R3,R4>
004316  100467                                                          MOV R0,-(SP)
004317  100461                                                          MOV R1,-(SP)
004320  100462                                                          MOV R2,-(SP)
004321  100463                                                          MOV R3,-(SP)
004322  100464                                                          MOV R4,-(SP)
004323  104203  000744                  MOV     #ST,R3                 ;R3 -> BYTE COUNT
004325  104137                          MOV     (R3),R0                ;R0 = BYTE COUNT
004326  103207  177400                  BIC     #HIBYTE,R0             ;STRIP OFF EXTRANEOUS
004330  104231                  1$:     MOV     (R3)+,R1               ;R1 = EVEN BYTE OF DATA
004331  103201  000377                  BIC     #LOBYTE,R1             ;STRIP OFF EXTRANEOUS
004333  117407                          DEC     R0                     ;DONE?
004334  014346                          BEQ     2$                     ;IF YES, GO STORE LAST BYTE
004335  104132                          MOV     (R3),R2                ;R2 = ODD BYTE
004336  103202  177400                  BIC     #HIBYTE,R2             ;STRIP OFF EXTRANEOUS
004340  101021                          BIS     R2,R1                  ;R1 HAS WHOLE WORD(BYTES REVERSED)
004341  110701                          SWAB    R1                     ;SWITCH BYTES
004342  100241                          MOV     R1,(R4)+               ;STORE DATA
004343  117407                          DEC     R0                     ;DONE?
004344  014350                          BEQ     3$                     ;IF SO, EXIT(EVEN # OF BYTES)
004345  004330                          BR      1$                     ;ELSE, CONTINUE
004346  110701                  2$:     SWAB    R1                     ;ODD # BYTES TO GET HERE
004347  100141                          MOV     R1,(R4)                ;STORE LAST BYTE
004350                          3$:     POP     <R4,R3,R2,R1,R0>
004350  104264                                                          MOV (SP)+,R4
004351  104263                                                          MOV (SP)+,R3
004352  104262                                                          MOV (SP)+,R2
004353  104261                                                          MOV (SP)+,R1
004354  104267                                                          MOV (SP)+,R0
004355  000000                          RETURN
```

```
                                        .SBTTL  GET BYTE COUNT
                                ;+
                                ;       ROUTINE NAME:   GETBCN
                                ;
                                ;       DESCRIPTION:
                                ;       DATA IS READ BY THE DM PROGRAM AFTER THE DIAGNOSE COMMAND.
                                ;       WITHIN THIS DATA, THE DRIVE SPECIFIES IF MORE DA'A NEEDS TO
                                ;       BE READ.  IF IT IS, THE BYTE COUNT MUST BE CONVERTED FROM
                                ;       ITS PRESENT FORM WHICH FOLLOWS THIS FORMAT:
                                ;
                                ;       BIT 15                                          BIT 0
                                ;       +-----------------------+-----------------------+
                                ;OUT.08:!       ASCII COUNT     !    BINARY COUNT       !
                                ;       +-----------------------+-----------------------+
                                ;
                                ;       THE TOTAL BYTE COUNT IS FOUND BY THIS FORMULA:
                                ;                       TBC = AC + (4*BC)
                                ;       OR THE TOTAL BYTE COUNT EQUALS THE ASCII COUNT PLUS FOUR
                                ;       TIMES THE BINARY COUNT.  THE BINARY COUNT IS THE NUMBER OF
                                ;       32 BIT BINARY VALUES INCLUDED IN THE REPORT.  THE ASCII
                                ;       COUNT IS THE NUMBER OF ASCII CHARACTERS INCLUDED IN THE
                                ;       REPORT.
                                ;
                                ;       INPUT:  OUT.08 = ASCII COUNT AND BINARY COUNT IN THE FORM STATED ABOVE
                                ;
                                ;       OUTPUT: RDM+3 HAS UPDATED BYTE COUNT.
                                ;
                                ;-
004356                          GETBCN: PUSH    <R0,R1,R2,R3,R4,R5>
004356  100467                                                          MOV R0,-(SP)
004357  100461                                                          MOV R1,-(SP)
004360  100462                                                          MOV R2,-(SP)
004361  100463                                                          MOV R3,-(SP)
004362  100464                                                          MOV R4,-(SP)
004363  100465                                                          MOV R5,-(SP)
004364  114007                          CLR     R0
004365  110207                          ROL     R0                      ;CLEAR CARRY
004366  104307  001510                  MOV     OUT.08,R0               ;GET BYTE COUNT
004370  103207  177400                  BIC     #HIBYTE,R0              ;R0 = BINARY COUNT= # 32 BIT WORDS
004372  110207                          ROL     R0                      ;
004373  110207                          ROL     R0                      ;R0 = BINARY COUNT IN BYTES
004374  104301  001510                  MOV     OUT.08,R1               ;R1 = BYTE COUNT
004376  110701                          SWAB    R1                      ;ASCII COUNT IN LO BYTE
004377  103201  177400                  BIC     #HIBYTE,R1              ;R1 = ASCII COUNT
004401  104012                          MOV     R1,R2                   ;SAVE IN R2 = ASCII COUNT
004402  105017                          ADD     R1,R0                   ;R0 = TOTAL BYTE COUNT
                                ; *** IS SIZE OF THE PACKET TOO BIG FOR BUFFER AVAILABLE?
004403  104201  000052                  MOV     #42.,R1                 ;42. IS THE MAXIMUM # OF BYTES LEFT
004405  106017                          CMP     R1,R0                   ;IS TOTAL BYTE COUNT > 42.?
004406  034473                          BPL     5$                      ;IF NOT, GO STORE TOTAL BYTE COUNT
                                ; *** SAVE OUT.RQ DATA
004407  104204  001501                  MOV     #OUT.01,R4              ;R4-> OUT.RQ DATA
004411  104205  001104                  MOV     #ST+140,R5              ;R5->SAVE AREA
004413  104241                  1$:     MOV     (R4)+,R1                ;SAVE
004414  100251                          MOV     R1,(R5)+                ;
004415  106204  001542                  CMP     #OUT.34,R4              ;DONE?
004417  054413                          BNE     1$                      ;IF NOT, CONTINUE
```

```
                                ;
004420                                  MSSGE   MSG2
004420  104300  001152  001501                                                  MOV     LUNIT,OUT.01
004423  104200  003342  001502                                                  MOV     #MSG2,OUT.02
004426  100467                                                                          MOV R0,-(SP)
004427  100461                                                                          MOV R1,-(SP)
004430  104207  060015                                                          MOV     #MESSAG,R0
004432  024643                                                                  CALL    HOSTRQ
004433  104261                                                                          MOV (SP)+,R1
004434  104267                                                                          MOV (SP)+,R0
                                ; *** RESTORE OUT.RQ DATA
004435  104204  001501                  MOV     #OUT.01,R4              ;R4-> OUT.RQ DATA
004437  104205  001104                  MOV     #ST+140,R5              ;R5->SAVE AREA
004441  104251          2$:     MOV     (R5)+,R1                ;RESTORE
004442  100241                  MOV     R1,(R4)+                ;
004443  106204  001542                  CMP     #OUT.34,R4              ;DONE?
004445  054441                  BNE     2$                      ;IF NOT, CONTINUE
                                ;
004446  104073                  MOV     R0,R3                   ;SAVE ASCII COUNT + 4*BINARY COUNT IN R3
004447  104207  000052                  MOV     #42.,R0                 ;ELSE, MAXIMUM BYTE COUNT IS STORED.
                                ; *** ADJUST ASCII COUNT IF TOO LARGE
004451  107023                  SUB     R2,R3                   ;R3 = BINARY COUNT
004452  106073                  CMP     R0,R3                   ;IF BINARY COUNT ALONE > 42.?
004453  074462                  BMI     3$                      ;IF SO, TO CLEAR ASCII COUNT AND ADJUST BINARY COUNT
                                ; *** ASCII COUNT IS > 42. TO GET HERE
004454  105032                  ADD     R3,R2                   ;R2 = TOTAL BYTE COUNT
004455  107072                  SUB     R0,R2                   ;R2 = ASCII COUNT ADJUSTMENT
004456  110702                  SWAB    R2                      ;PUT ASCII COUNT ADJUSTMENT IN UPPER BYTE
004457  107020  001510                  SUB     R2,OUT.08               ;SUBTRACT FROM OUT.08 FOR HOST
004461  004473                  BR      5$                      ;EXIT
                                ; *** ADJUST BINARY COUNT IF TOO LARGE/CLEAR OUT ASCII COUNT
004462  103200  177400  001510  3$:     BIC     #HIBYTE,OUT.08          ;CLEAR HI BYTE OF OUT.08
004465  107073                  SUB     R0,R3                   ;R3 = BINARY COUNT ALONE
004466  117400  001510          4$:     DEC     OUT.08                  ;ADJUST BINARY COUNT
004470  107203  000004                  SUB     #4,R3
004472  034466                  BPL     4$
                                ; *** STORE IN READ MEMORY PACKET AND EXIT
004473                          5$:
004473  104070  001431                  MOV     R0,RDM+3                ;STORE IN READ MEMORY PACKET
004475                          POP     <R5,R4,R3,R2,R1,R0>
004475  104265                                                                  MOV (SP)+,R5
004476  104264                                                                  MOV (SP)+,R4
004477  104263                                                                  MOV (SP)+,R3
004500  104262                                                                  MOV (SP)+,R2
004501  104261                                                                  MOV (SP)+,R1
004502  104267                                                                  MOV (SP)+,R0
004503  000000                  RETURN
```

```
                                                .SBTTL  TYPE WHAT KIND OF RECEIVE ERROR
                                        ;:+
                                        ;       TYPE ERROR
                                        ;
                                        ;       DESCRIPTION:
                                        ;       THIS ROUTINE PRINTS A REPORT TO TELL WHAT KIND OF ERROR OCCURED IF A
                                        ;       RECEIVE XFC DID NOT SUCCESSFULLY COMPLETE.
                                        ;
                                        ;       INPUT:  R3 = ERROR VALUE FROM THE XFC SAVED IN TALKER ROUTINE SHIFTED LEFT ONCE
                                        ;                       IF = 1 TIMEOUT
                                        ;                       IF = 2 1ST WORD WAS NOT A START FRAME
                                        ;                       IF = 4 FRAMING ERROR ON SDI LEVEL 0 READ
                                        ;                       IF = 10 CHECKSUM ERROR ON SDI LEVEL 0 READ
                                        ;                       IF = 20 BUFFER SIZE WAS SMALLER THAN RESPONSE
                                        ;:-
004504  110603                 TYPERR: ROR     R3                      ;READJUST R3
004505  114001                         CLR     R1
004506  110201                         ROL     R1                      ;CLEAR CARRY
004507  104031                         MOV     R3,R1                   ;STORE IN R1
004510  110601                         ROR     R1                      ;ERROR?
004511  044515                         BCC     1$                      ;IF NOT, CONTINUE
004512  104207  003374                 MOV     #MSR0,R0                ;TIMEOUT
004514  004543                         BR      6$                      ;EXIT
004515  110601                 1$:     ROR     R1                      ;ERROR?
004516  044522                         BCC     2$                      ;IF NOT, CONTINUE
004517  104207  003423                 MOV     #MSR1,R0                ;1ST WORD WAS NOT A START FRAME
004521  004543                         BR      6$                      ;EXIT
004522  110601                 2$:     ROR     R1                      ;ERROR?
004523  044527                         BCC     3$                      ;IF NOT, CONTINUE
004524  104207  003453                 MOV     #MSR2,R0                ;FRAMING ERROR ON SDI LEVEL 0 READ
004526  004543                         BR      6$                      ;EXIT
004527  110601                 3$:     ROR     R1                      ;ERROR?
004530  044534                         BCC     4$                      ;IF NOT, CONTINUE
004531  104207  003514                 MOV     #MSR3,R0                ;CHECKSUM ERROR ON SDI LEVEL 0 READ
004533  004543                         BR      6$                      ;EXIT
004534  110601                 4$:     ROR     R1                      ;ERROR?
004535  044541                         BCC     5$                      ;IF NOT, CONTINUE
004536  104207  003555                 MOV     #MSR4,R0                ;BUFFER SIZE SMALLER THAN RESPONSE
004540  004543                         BR      6$                      ;EXIT
004541  104207  003612         5$:     MOV     #MSR5,R0                ;ERROR WASN'T PROPERLY SPECIFIED
004543  000000                 6$:     RETURN
```

```
                                        .SBTTL  DIVIDE BY OCTAL 50 AND FIND ASCII EQUIVALENT
                                ;:+
                                ;       DIVIDE BY 50
                                ;
                                ;       DESCRIPTION:    RAD 50 VALUE IN R4 IS DIVIDED BY 50(OCTAL)
                                ;                       TO STRIP OFF A SINGLE CHARACTER INTO R0.
                                ;                       JUMP TO FNDASC TO FIND THE ASCII EQUIVALENT
                                ;
                                ;       INPUT:  R4 HAS RAD50 VALUE
                                ;
                                ;       OUTPUT: R0 HAS ASCII CHARACTER
                                ;:-
004544                          DIV50:  PUSH    <R1>
004544  100461                                                                  MOV R1,-(SP)
004545  114001                          CLR     R1                      ;R1 = QUO
004546  115401                  1$:     INC     R1                      ;KEEP TRACK OF # OF LOOPS
004547  107204  000050                  SUB     #50,R4                  ;DONE?
004551  034546                          BPL     1$                      ;IF NOT, LOOP
004552  105204  000050                  ADD     #50,R4                  ;R4 = REMAINDER
004554  104047                          MOV     R4,R0                   ;STORE REMAINDER IN R0
004555  117401                          DEC     R1                      ;R1 HAS 1 MORE THAN IT SHOULD
004556  104014                          MOV     R1,R4                   ;QUO IN R4
004557                                  POP     <R1>
004557  104261                                                                  MOV (SP)+,R1

                                ;:+
                                ;       FIND ASCII
                                ;
                                ;       INPUT:  R0 HAS RAD 50 CHARACTER
                                ;:-
004560  115007                  FNDASC: TST     R0                      ;IS = 0?
004561  054565                          BNE     2$                      ;IF NOT, CONTINUE
004562  104207  000040                  MOV     #40,R0                  ;IF SO, SPACE SET
004564  000000                          RETURN
004565  106207  000032          2$:     CMP     #32,R0                  ; > LARGEST LETTER
004567  074573                          BMI     3$                      ;IF SO, CONTINUE
004570  105207  000100                  ADD     #100,R0                 ;SET ASCII LETTER
004572  000000                          RETURN
004573  106207  000033          3$:     CMP     #33,R0                  ; = 33?
004575  054601                          BNE     4$                      ;IF NOT, CONTINUE
004576  104207  000044                  MOV     #44,R0                  ;SET '$'
004600  000000                          RETURN
004601  106207  000034          4$:     CMP     #34,R0                  ; = 34?
004603  054607                          BNE     5$                      ;IF NOT, CONTINUE
004604  104207  000056                  MOV     #56,R0                  ;SET '.'
004606  000000                          RETURN
004607  105207  000022          5$:     ADD     #22,R0                  ;ELSE ADD 22(OCTAL)FOR NUMERAL CHARACTER
004611  000000                          RETURN
```

```
                                        .SBTTL  RDSTAT - GET DRIVE'S REAL TIME DRIVE STATE
004612                                  RDSTAT:
                                        ;
                                        ;RETURN DRIVE STATUS
                                        ;STATUS RETURNED IN DM REGISTER 1
                                        ;
004612                                          PUSH    <R3,R0>                 ; SAVE R3 AND R0
004612  100463                                                                          MOV R3,-(SP)
004613  100467                                                                          MOV R0,-(SP)
004614  104203  000003                         MOV     #3,R3                   ; ALLOW ONLY 3 ERRORS
004616  060007                  STATLP: XFC     STATUS                  ; GET DRIVE'S STATUS
004617  103201  014000                  BIC     #14000,R1               ; CLEAR ERROR PASSING BITS
004621  102201  000400                  BIT     #XMTERR,R1              ; CHECK XMIT ERRORS
004623  014631                          BEQ     STATOK                  ; IF NO ERRORS, BRANCH
004624  117403                          DEC     R3                      ; DECREMENT TRANSMIT ERROR COUNT
004625  054616                          BNE     STATLP                  ; IF ERROR COUNT INCOMPLETE, BRANCH
004626  104201  010000                  MOV     #10000,R1               ; FLAG AS TRANSMIT ERROR
004630  004640                          BR      STATEX                  ; BRANCH
004631  102201  000004          STATOK: BIT     #RCVERR,R1              ; RECIEVER ERRORS
004633  054640                          BNE     STATEX                  ; IF VALID, BRANCH
004634  117403                          DEC     R3                      ; DECREMENT ERROR COUNT
004635  054616                          BNE     STATLP                  ; IF ERROR COUNT NON-ZERO, BRANCH
004636  104201  014000                  MOV     #14000,R1               ; FLAG AS INVALID STATUS ERROR
004640                          STATEX: POP     <R0,R3>                 ; RESTORE R0, R3
004640  104267                                                                          MOV (SP)+,R0
004641  104263                                                                          MOV (SP)+,R3
004642  000000                          RETURN                          ; RETURN TO CALLING MODULE
```

```
                                .SBTTL  HOSTRQ - HOST REQUEST - REPORT ERRORS, MEGABYTES TRANSFERRED, ETC.
004643                          HOSTRQ:
                                ;
                                ;SEND REQUEST BUFFER TO HOST AND WAIT FOR RESPONSE.
                                ;CLEAR ARGUMENT AREA OF OUT BUFFER IN PREPARATION
                                ;FOR NEXT HOSTRQ CALL.
                                ;
                                ;INPUTS:
                                ;       R0 - HOST REQUEST NUMBER
                                ;       OUT BUFFER LOADED WITH DATA

004643                          PUSH    <R0,R1,R2>
004643  100467                                                                  MOV R0,-(SP)
004644  100461                                                                  MOV R1,-(SP)
004645  100462                                                                  MOV R2,-(SP)
004646  104070  001500                 MOV     R0,OUT.RQ               ; STORE REQUEST NUMBER IN BUFFER
004650  104300  001152  001501 SNDAGN: MOV     LUNIT,OUT.01
004653  104207  001500                 MOV     #OUT.RQ,R0              ; SEND BUFFER TO HOST
004655  104201  000043                 MOV     #BUFSIZ,R1
004657  060016                         XFC     MRD
004660  115001                         TST     R1                      ; CHECK FOR ERROR
004661  054650                         BNE     SNDAGN                  ; IF ERROR, TRY AGAIN
004662  104207  001543                 MOV     #IN.RQ,R0               ; WAIT FOR RESPONSE FROM HOST
004664  104201  000043                 MOV     #BUFSIZ,R1
004666  060017                         XFC     MWR
004667  104200  177777  001500        MOV     #-1,OUT.RQ              ; MAKE REQUEST ILLEGAL
004672  104207  001501                 MOV     #OUT.01,R0              ; CLEAR ARGUMENT WORDS IN BUFFER
004674  104201  000042                 MOV     #BUFSIZ-1,R1            ; CLEAR ENTIRE BUFFER
004676  114002                         CLR     R2
004677  100272                 CLRBUF: MOV     R2,(R0)+
004700  117401                         DEC     R1
004701  034677                         BPL     CLRBUF
004702                         POP     <R2,R1,R0>
004702  104262                                                                  MOV (SP)+,R2
004703  104261                                                                  MOV (SP)+,R1
004704  104267                                                                  MOV (SP)+,R0
004705  000000                 RETURN
```

```
                                        ;
                                        ;       TALKER SENDS AND RECEIVES AN SDI INTERCHANGE
                                        ;
                                        ;INPUTS:
                                        ;       R2 - SDI INTERCONNECT
                                        ;       R3 - POINTER TO COMMAND TABLE CONTAINING APPROPRIATE COMMAND
                                        ;       SDILTO/SDISTO SDI LONG AND SHORT TIMEOUTS, RESPECTIVELY
                                        ;OUTPUTS:
                                        ;       R0 - RETURN OP CODE FROM UNIT
                                        ;       R3 - ERROR CODE - 0 = NO ERROR, 1 = RECEIVE ERROR, 100001 = SEND ERROR
                                        ;
004706                          TALKER: PUSH    <R1,R4>         ; SAVE REGISTERS
004706  100461                                                                  MOV R1,-(SP)
004707  100464                                                                  MOV R4,-(SP)
004710  104237                          MOV     (R3)+,R0        ; SET ADR OF SDI COMMAND BUFFER
004711  104231                          MOV     (R3)+,R1        ; SET BUFFER LENGTH
004712  060004                          XFC     SEND            ; SEND COMMAND
004713  115001                          TST     R1              ; DID UNIT ACCEPT COMMAND
004714  014720                          BEQ     TALK1A          ; IF SO, BRANCH
004715  104203  100001                  MOV     #100001,R3      ; FLAG AS SEND ERROR
004717  004766                          BR      TALK2B          ; BRANCH TO EXIT
004720  106203  001454          TALK1A: CMP     #LONG,R3        ; SEE IF LONG TIMEOUT TO BE USED
004722  074726                          BMI     1$              ; IF SO, BRANCH
004723  104304  001154                  MOV     SDISTO,R4       ; SET UP SHORT TIMEOUT
004725  004730                          BR      TALK1B          ; BRANCH
004726  104304  001155          1$:     MOV     SDILTO,R4       ; SET UP LONG TIMEOUT
004730                          TALK1B:
004730  115000  001151                  TST     SENDHR          ; DO WE SEND HOSTRQ?
004732  014745                          BEQ     1$              ; IN NOT, CONTINUE
004733  104300  001152  001501          MOV     LUNIT,OUT.01    ; SEND UNIT NUMBER
004736  114000  001502                  CLR     OUT.02          ; NO MEGABYTES READ
004740  114000  001503                  CLR     OUT.03          ; NO MEGABYTES WRITTEN
004742  104207  060011                  MOV     #T4MXFR,R0      ; SEND SOMETHING TO HOST
004744  024643                          CALL    HOSTRQ          ;  SO HOST WON'T TIMEOUT DM PROG
004745  104137                  1$:     MOV     (R3),R0         ; SET DATA BUFFER ADDRESS
004746  104631  000001                  MOV     1(R3),R1        ; SET BUFFER LENGTH
004750  060005                          XFC     RCV             ; SEND RECEIVE SDI COMMAND
004751  115001                          TST     R1              ; DID ERROR OCCUR
004752  014765                          BEQ     TALK2A          ; IF NOT, BRANCH
004753  106201  000001                  CMP     #1,R1           ; SEE IF TIMEOUT
004755  014760                          BEQ     2$              ; IF SO, BRANCH
004756  104013                          MOV     R1,R3           ; MOVE ERROR TYPE TO R3 FOR REPORTING
004757  004766                          BR      TALK2B          ; EXIT
004760  117404                  2$:     DEC     R4              ; DECREMENT TIMEOUT VALUE
004761  054730                          BNE     TALK1B          ; IF NOT TIMEOUT, BRANCH
004762  104203  000001                  MOV     #1,R3           ; FLAG AS RECIEVE ERROR
004764  004766                          BR      TALK2B          ; BRANCH TO EXIT
004765  114003                  TALK2A: CLR     R3              ; FLAG AS NO ERRORS
004766                          TALK2B: POP     <R4,R1>         ; RESTORE R4, R1
004766  104264                                                                  MOV (SP)+,R4
004767  104261                                                                  MOV (SP)+,R1
004770  000000                          RETURN
```

```
UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42 PAGE 57
HOSTRQ - HOST REQUEST - REPORT ERRORS, MEGABYTES TRANSFERRED,

      1                                 ;
      2                                 ;XOR
      3                                 ;
      4                                 ;PERFORM XOR LOGIC FUNCTION ON TWO REGISTERS
      5                                 ;INPUTS:
      6                                 ;       R1, R2 - DATA TO BE XOR'ED
      7                                 ;OUTPUTS:
      8                                 ;       R1 - UNCHANGED
      9                                 ;       R2 - XOR OF TWO INPUTS
     10
     11 004771                          XOR:    PUSH R3                         ;SAVE R3
        004771  100463                                                                  MOV R3,-(SP)
     12 004772  104013                          MOV R1,R3
     13 004773  103023                          BIC R2,R3
     14 004774  103012                          BIC R1,R2
     15 004775  101032                          BIS R3,R2
     16 004776                                  POP R3                          ;RESTORE R3
        004776  104263                                                                  MOV (SP)+,R3
     17 004777  115002                          TST R2                          ;SET CONDITION CODES
     18 005000  000000                          RETURN
```

```
UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42 PAGE 58
HOSTRQ - HOST REQUEST - REPORT ERRORS, MEGABYTES TRANSFERRED,

      1 005001                          TO:
      2                                 ;
      3                                 ;       CALCULATE THE TIMEOUT IN 9SEC INTERVALS (SDI RECEIVE XFC TAKES
      4                                 ;       9 SEC)
      5                                 ;
      6 005001  104201  000001                  MOV     #1,R1           ; SET UP LOG2 SHIFTER
      7 005003  110201                  1$:     ROL     R1              ; DOUBLE THE TIMEOUT VALUE
      8 005004  103201  000001                  BIC     #1,R1           ; CLEAR THE LOW BIT
      9 005006  117407                          DEC     R0              ; DECREMENT COUNT
     10 005007  055003                          BNE     1$              ; IF COUNT INCOMPLETE, BRANCH
     11 005010  114007                          CLR     R0              ; CLEAR 9SEC COUNT
     12 005011  115407                  2$:     INC     R0              ; INCREMENT 9 SEC COUNT
     13 005012  107201  000011                  SUB     #9.,R1          ; SUBTRACT 9 SEC FROM TIMEOUT
     14 005014  035011                          BPL     2$              ; IF MORE TIME TO GO, BRANCH
     15 005015  000000                          RETURN                  ; RETURN TO CALLING PROGRAM
```

HOSTRQ - HOST REQUEST - REPORT ERRORS, MEGABYTES TRANSFERRED,

```
                                ;RERROR
                                ;
                                ;REPORT ERROR TO HOST PROGRAM
                                ;THIS ROUTINE IS CALLED BY THE ERROR MACROS:
                                ;ERRSF, ERRDF, ERRHRD AND ERRSFT
                                ;
005016                          RERROR: PUSH R0                         ;SAVE ONE REGISTER
005016  100467                                                                 MOV R0,-(SP)
005017  104067                          MOV SP,R0                       ;GET STACK POINTER
005020                                  PUSH <R1,R2,R3,R4>              ;SAVE MORE REGISTERS
005020  100461                                                                 MOV R1,-(SP)
005021  100462                                                                 MOV R2,-(SP)
005022  100463                                                                 MOV R3,-(SP)
005023  100464                                                                 MOV R4,-(SP)
005024  115407                          INC R0                          ;CHANGE SAVED STACK POINTER TO POINT TO
                                                                        ; ADDRESS OF LOCATION AFTER CALL
005025  104271                          MOV (R0)+,R1                    ;GET RETURN PC
005026  104202  001501                  MOV #OUT.01,R2                  ;GET ADDRESS OF WHERE TO PUT DATA
005030  117401                          DEC R1                          ;REDUCE TO PC OF ERROR CALL
005031  100221                          MOV R1,(R2)+                    ;PUT PC IN OUT BUFFER
005032  115401                          INC R1                          ;GET BACK TO RETURN PC
005033  104213                          MOV (R1)+,R3                    ;GET ERROR NUMBER AND TYPE
005034  100223                          MOV R3,(R2)+                    ;PUT IN BUFFER
005035  104303  001152                  MOV LUNIT,R3                    ;PUT UNIT NUMBER IN BUFFER
005037  100223                          MOV R3,(R2)+
005040  104113                          MOV (R1),R3                     ;GET MESSAGE POINTER
005041  103203  170000                  BIC #^C007777,R3                ;CLEAR OTHER BITS
005043  100223                          MOV R3,(R2)+                    ;PUT IN OUT BUFFER
005044  104214                          MOV (R1)+,R4                    ;GET COUNT OF PARAMETERS
                                                                        ;R1 IS NOW POINTING TO INSTRUCTION AFTER ERROR CALL
005045  110704                          SWAB R4                         ;EXTRACT NUMBER OF PARAMETERS FROM ERROR MACRO
005046  110604                          ROR R4
005047  110604                          ROR R4
005050  110604                          ROR R4
005051  110604                          ROR R4
005052  103204  177760                  BIC #177760,R4
005054  104040  005075                  MOV R4,SPADJU+1                 ;SAVE FOR LATER ADJUSTMENT OF STACK POINTER
005056  015063                          BEQ RERRCA                      ;BRANCH IF NO PARAMETERS
005057  104273                  RERRPA: MOV (R0)+,R3                    ;GET PARAMETER
005060  100223                          MOV R3,(R2)+                    ;STORE PARAMETER IN OUT BUFFER
005061  117404                          DEC R4                          ;COUNT THE PARAMETERS
005062  055057                          BNE RERRPA                      ;GET NEXT IF MORE
005063  100471                  RERRCA: MOV R1,-(R0)                    ;PUT RETURN ADDRESS ON STACK
005064  104207  060013                  MOV #ERRMES,R0                  ;SEND ERROR PACKET TO HOST PROGRAM
005066  024643                          CALL HOSTRQ
005067                                  POP <R4,R3,R2,R1,R0>            ;RESTORE REGISTERS
005067  104264                                                                 MOV (SP)+,R4
005070  104263                                                                 MOV (SP)+,R3
005071  104262                                                                 MOV (SP)+,R2
005072  104261                                                                 MOV (SP)+,R1
005073  104267                                                                 MOV (SP)+,R0
005074  105206  000000          SPADJU: ADD #0,SP                       ;ADJUST STACK OVER PARAMETERS
                                                                        ;VALUE CHANGED ABOVE
005076  000000                          RETURN
```

```
                                        ;MESSAGE STORAGE OVERLAY
                                        
005077                                  FREE:                                   ; FREE AREA STARTS HERE
005077                                          DMOVLY MS,0
005077  073442                                  .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                                .NLIST BEX
000000      042     110     117  MS2000: .ASCII\'HOST SPECIFIED UNIT #'D16'' THAT CAN'T BE FOUND.'N\
000031      042     124     105          .ASCII\'TEST2 RESTARING'N\
000042      000                          .BYTE   0
000043      042     103     101  MS2001: .ASCII\'CANNOT RECEIVE VALID DRIVE STATE FROM DRIVE AFTER DRIVE WAS INITED'N\
000105      042     103     110          .ASCII\'CHECK IF DRIVE IS POWERED ON.'N\
000125      000                          .BYTE 0
000126      042     104     122  MS2002: .ASCII\'DRIVE STATE RECEIVED HAS BAD PARITY AFTER DRIVE WAS INITED'N\
000164      000                          .BYTE 0
000165      042     104     122  MS2003: .ASCII\'DRIVE IS NOT ASSERTING RECEIVER READY IN DRIVE STATE AFTER DRIVE WAS INITED
000234      000                          .BYTE 0
000235      042     124     111  MS2004: .ASCII\'TIME-OUT ON SEND OF ECHO COMMAND TO DRIVE'N\
000263      042     040     040          .ASCII\'  ECHO DATA  'H8N\
000274      000                          .BYTE 0
000275      042     105     122  MS2005: .ASCII\'ERROR DURING RECEIVE OF ECHO RESPONSE FROM DRIVE'N\
000326      042     040     040          .ASCII\'  ECHO DATA  'H8N\
000337      000                          .BYTE 0
000340      042     105     103  MS2006: .ASCII\'ECHO COMMAND RESPONDED WITH DIFFERENT DATA'N\
000366      042     040     040          .ASCII\'  ECHO DATA SENT''S6H16N\
000402      042     040     040          .ASCII\'  ECHO DATA RECEIVED  'H16N\
000420      000                          .BYTE 0
000421      042     105     122  MS2007: .ASCII\'ERROR BIT SET IN GET STATUS RESPONSE AFTER DRIVE CLEAR COMMAND'N\
000461      042     040     040          .ASCII\'  GET STATUS RESPONSE  'NR1\
000477      000                          .BYTE 0
000500      042     124     111  MS2008: .ASCII\'TIME-OUT ON SEND OF ONLINE COMMAND TO DRIVE'N\
000527      000                          .BYTE 0
000530      042     105     122  MS2009: .ASCII\'ERROR DURING RECEIVE OF ONLINE RESPONSE FROM DRIVE'R1\
000563      000                          .BYTE 0
000564      042     117     116  MS2010: .ASCII\'ONLINE COMMAND WAS UNSUCCESSFUL'NR1\
000606      000                          .BYTE 0
000607      042     117     116  MS2011: .ASCII\'ONLINE COMMAND DID NOT RETURN EXPECTED RESPONSE CODE'N\
000642      042     040     040          .ASCII\'  EXPECTED RESPONSE  'H8N\
000657      042     040     040          .ASCII\'  ACTUAL RESPONSE''S4H8N\
000673      000                          .BYTE 0
000674      042     124     111  MS2012: .ASCII\'TIME-OUT ON SEND OF GET UNIT CHARACTERISTICS COMMAND TO DRIVE'N\
000734      000                          .BYTE   0
000735      042     105     122  MS2013: .ASCII\'ERROR DURING RECEIVE OF GET UNIT CHARACTERISTICS COMMAND FROM DRIVE'R1\
001000      000                          .BYTE   0
001001      042     107     105  MS2014: .ASCII\'GET UNIT CHARACTERISTICS COMMAND WAS UNSUCCESSFUL'NR1\
001034      000                          .BYTE   0
001035      042     107     105  MS2015: .ASCII\'GET UNIT CHARACTERISTICS COMMAND DID NOT RETURN EXPECTED RESPONSE CODE'N\
001101      042     040     040          .ASCII\'  EXPECTED RESPONSE  'H8N\
001116      042     040     040          .ASCII\'  ACTUAL RESPONSE''S4H8N\
001132      000                          .BYTE   0
001133      042     040     040  MS2016: .ASCII\'  HOST PROGRAM GAVE DM CODE IMPROPER DATA'N\
001161      042     040     040          .ASCII\'  EXPECTED VALUE SHOULD BE BETWEEN 0 AND 3'N\
001207      042     040     040          .ASCII\'  ACTUAL VALUE WAS 'D8N\
001223      000                          .BYTE 0
001224      042     124     111  MS2017: .ASCII\'TIME-OUT ON SEND OF DIAGNOSE COMMAND TO DRIVE'N\
001254      000                          .BYTE 0
001255      042     105     122  MS2018: .ASCII\'ERROR DURING RECEIVE OF DIAGNOSE RESPONSE FROM DRIVE'R1\
001311      000                          .BYTE 0
```

```
UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42 PAGE 60-1
HOSTRQ - HOST REQUEST - REPORT ERRORS, MEGABYTES TRANSFERRED,

     57 001312      042      104      111  MS2019: .ASCII\'DIAGNOSE COMMAND WAS UNSUCCESSFUL'NR1\
     58 001335      000                            .BYTE   0
     59 001336      042      104      111  MS2020: .ASCII\'DIAGNOSE COMMAND DID NOT RETURN EXPECTED RESPONSE CODE'N\
     60 001372      042      040      040          .ASCII\'' EXPECTED RESPONSE  '%H8N\
     61 001407      042      040      040          .ASCII\'' ACTUAL RESPONSE''S4%H8N\
     62 001423      000                            .BYTE 0
     63 001424      042      104      122  MS2021: .ASCII\'DRIVE DIAGNOSTIC REPORTS A HARD ERROR'N\
     64 001450      042      040      040          .ASCII\'' TEST NUMBER  '%H16N\
     65 001462      042      040      040          .ASCII\'' DRIVE TYPE   '%H8N\
     66 001474      042      040      040          .ASCII\'' ERROR NUMBER '%H16N\
     67 001507      122      062                   .ASCII\R2\
     68 001510      000                            .BYTE 0
     69 001511      042      110      117  MS2022: .ASCII\'HOST PROGRAM DOWN LINE LOADED A DIAGNOSTIC WITH A ZERO BYTE COUNT'N\
     70 001553      000                            .BYTE 0
     71 001554      042      104      111  MS2023: .ASCII\'DIAGNOSTIC ''A6'' REQUESTED BY THE DRIVE COULD NOT BE SUPPLIED BY HOST.'N\
     72 001620      000                            .BYTE   0
     73 001621      042      124      111  MS2024: .ASCII\'TIME-OUT ON SEND OF MEMORY READ COMMAND TO DRIVE'N\
     74 001652      000                            .BYTE 0
     75 001653      042      105      122  MS2025: .ASCII\'ERROR DURING RECEIVE OF MEMORY READ RESPONSE FROM DRIVE'NR1\
     76 001711      000                            .BYTE 0
     77 001712      042      115      105  MS2026: .ASCII\'MEMORY READ COMMAND WAS UNSUCCESSFUL'NR1\
     78 001736      000                            .BYTE   0
     79 001737      042      115      105  MS2027: .ASCII\'MEMORY READ COMMAND DID NOT RETURN EXPECTED RESPONSE CODE'N\
     80 001775      042      040      040          .ASCII\'' EXPECTED RESPONSE  '%H8N\
     81 002012      042      040      040          .ASCII\'' ACTUAL RESPONSE''S4%H8N\
     82 002026      000                            .BYTE 0
     83 002027      042      124      111  MS2028: .ASCII\'TIME-OUT ON SEND OF MEMORY WRITE COMMAND TO DRIVE'N\
     84 002061      000                            .BYTE 0
     85 002062      042      105      122  MS2029: .ASCII\'ERROR DURING RECEIVE OF MEMORY WRITE RESPONSE FROM DRIVE'R1\
     86 002120      000                            .BYTE 0
     87 002121      042      115      105  MS2030: .ASCII\'MEMORY WRITE COMMAND WAS UNSUCCESSFUL'NR1\
     88 002146      000                            .BYTE   0
     89 002147      042      115      105  MS2031: .ASCII\'MEMORY WRITE COMMAND DID NOT RETURN EXPECTED RESPONSE CODE'N\
     90 002205      042      040      040          .ASCII\'' EXPECTED RESPONSE  '%H8N\
     91 002222      042      040      040          .ASCII\'' ACTUAL RESPONSE''S4%H8N\
     92 002236      000                            .BYTE   0
     93 002237      042      124      111  MS2032: .ASCII\'TIME-OUT ON SEND OF RUN COMMAND TO DRIVE'N\
     94 002264      000                            .BYTE 0
     95 002265      042      105      122  MS2033: .ASCII\'ERROR DURING RECEIVE OF RUN RESPONSE FROM DRIVE'R1\
     96 002316      000                            .BYTE 0
     97 002317      042      122      125  MS2034: .ASCII\'RUN COMMAND WAS UNSUCCESSFUL'NR1\
     98 002337      000                            .BYTE 0
     99 002340      042      122      125  MS2035: .ASCII\'RUN COMMAND DID NOT RETURN EXPECTED RESPONSE CODE'N\
    100 002372      042      040      040          .ASCII\'' EXPECTED RESPONSE  '%H8N\
    101 002407      042      040      040          .ASCII\'' ACTUAL RESPONSE''S4%H8N\
    102 002423      000                            .BYTE 0
    103 002424      042      124      111  MS2036: .ASCII\'TIME-OUT ON SEND OF RECALIBRATE COMMAND TO DRIVE'N\
    104 002455      000                            .BYTE 0
    105 002456      042      105      122  MS2037: .ASCII\'ERROR DURING RECEIVE OF RECALIBRATE RESPONSE FROM DRIVE'R1\
    106 002513      000                            .BYTE 0
    107 002514      042      122      105  MS2038: .ASCII\'RECALIBRATE COMMAND WAS UNSUCCESSFUL'NR1\
    108 002540      000                            .BYTE 0
    109 002541      042      122      105  MS2039: .ASCII\'RECALIBRATE COMMAND DID NOT RETURN EXPECTED RESPONSE CODE'N\
    110 002577      042      040      040          .ASCII\'' EXPECTED RESPONSE  '%H8N\
    111 002614      042      040      040          .ASCII\'' ACTUAL RESPONSE''S4%H8N\
    112 002630      000                            .BYTE 0
    113 002631      042      124      111  MS2040: .ASCII\'TIME-OUT ON SEND OF GET STATUS COMMAND TO DRIVE'N\
```

```
114 002662  000                                 .BYTE 0
115 002663  042  105  122  MS2041:  .ASCII\'ERROR DURING RECEIVE OF GET STATUS RESPONSE FROM DRIVE'R1\
116 002720  000                                 .BYTE 0
117 002721  042  107  105  MS2042:  .ASCII\'GET STATUS COMMAND WAS UNSUCCESSFUL'NR1\
118 002745  000                                 .BYTE 0
119 002746  042  107  105  MS2043:  .ASCII\'GET STATUS COMMAND DID NOT RETURN EXPECTED RESPONSE CODE'N\
120 003003  042  040  040           .ASCII\"  EXPECTED RESPONSE  '"H8N\
121 003020  042  040  040           .ASCII\"  ACTUAL RESPONSE'"S4H8N\
122 003034  000                                 .BYTE 0
123 003035  042  124  111  MS2044:  .ASCII\'TIME-OUT ON SEND OF DRIVE CLEAR COMMAND TO DRIVE'N\
124 003066  000                                 .BYTE 0
125 003067  042  105  122  MS2045:  .ASCII\'ERROR DURING RECEIVE OF DRIVE CLEAR RESPONSE FROM DRIVE'R1\
126 003124  000                                 .BYTE 0
127 003125  042  104  122  MS2046:  .ASCII\'DRIVE CLEAR COMMAND WAS UNSUCCESSFUL'NR1\
128 003151  000                                 .BYTE 0
129 003152  042  104  122  MS2047:  .ASCII\'DRIVE CLEAR COMMAND DID NOT RETURN EXPECTED RESPONSE CODE'N\
130 003210  042  040  040           .ASCII\"  EXPECTED RESPONSE  '"H8N\
131 003225  042  040  040           .ASCII\"  ACTUAL RESPONSE'"S4H8N\
132 003241  000                                 .BYTE 0
133 003242  101  064  116  MSG1:    .ASCII\A4N\
134 003243  042  111  116           .ASCII\'INFORMATION SENT BACK FROM THE DRIVE IS BEING PRESENTED.'N\
135 003301  042  040  040           .ASCII\"  TEST NUMBER  '"H16N\
136 003313  042  040  040           .ASCII\"  DRIVE TYPE   '"H8N\
137 003325  042  040  040           .ASCII\"  ERROR NUMBER '"H16N\
138 003340  122  062                .ASCII\R2\
139 003341  000                                 .BYTE 0
140 003342  042  106  117  MSG2:    .ASCII\'FOLLOWING REPORT HAS BEEN TRUNCATED DUE TO SIZE'N\
141 003373  000                                 .BYTE 0
142 003374  116  042  124  MSR0:    .ASCII\N'"TIMEOUT ERROR OCCURED DURING RECEIVE XFC'N\
143 003422  000                                 .BYTE   0
144 003423  116  042  061  MSR1:    .ASCII\N'"1ST WORD NOT START FRAME DURING RECEIVE XFC'N\
145 003452  000                                 .BYTE   0
146 003453  116                    MSR2:    .ASCII\N\
147 003453  042  106  122           .ASCII\'FRAMING ERROR OCCURED ON SDI LEVEL 0 READ DURING RECEIVE XFC'N\
148 003513  000                                 .BYTE   0
149 003514  116                    MSR3:    .ASCII\N\
150 003514  042  103  110           .ASCII\'CHECKSUM ERROR OCCURED ON SDI LEVEL 0 READ DURING RECEIVE XFC'N\
151 003554  000                                 .BYTE   0
152 003555  116                    MSR4:    .ASCII\N\
153 003555  042  102  125           .ASCII\'BUFFER SIZE SMALLER THEN RESPONSE DURING RECEIVE XFC'N\
154 003611  000                                 .BYTE   0
155 003612  116                    MSR5:    .ASCII\N\
156 003612  042  103  117           .ASCII\'CODE FROM RECEIVE XFC WAS UNINTELLIGIBLE FROM SUBSYSTEM  '"H16N\
157 003652  000                                 .BYTE   0
158 003653  042  125  116  ER5000:  .ASCII  \'UNABLE TO FIND REQUESTED DRIVE FOR TESTING'N\
159 003701  042  124  110           .ASCII  \'THE FOLLOWING IS VISIBLE ON THE PORTS'N\
160 003725  042  125  104           .ASCII  \'UDA PORT 0 -- '"R1\
161 003736  042  125  104           .ASCII  \'UDA PORT 1 -- '"R1\
162 003747  042  125  104           .ASCII  \'UDA PORT 2 -- '"R1\
163 003760  042  125  104           .ASCII  \'UDA PORT 3 -- '"R1\
164 003771  000                                 .BYTE   0
165 003772  042  116  117  SER10:   .ASCII  \'NO DRIVE ATTACHED'N\
166 004004  000                                 .BYTE   0
167 004005  042  122  103  SER11:   .ASCII  \'RCVR RDY NEVER ASSERTED'N\
168 004022  000                                 .BYTE   0
169 004023  042  124  111  SER12:   .ASCII  \'TIMEOUT OF SEND'N\
170 004034  000                                 .BYTE   0
```

```
    171 004035      042     124     111  SER13:  .ASCII  \"TIMEOUT OF RECEIVE'N\
    172 004047      000                          .BYTE   0
    173 004050      042     106     111  SER14:  .ASCII  \"FIRST WORD RECEIVED WAS NOT START FRAME'N\
    174 004075      000                          .BYTE   0
    175 004076      042     106     122  SER15:  .ASCII  \"FRAMING ERROR ON LEVEL 0 RECEIVE'N\
    176 004117      000                          .BYTE   0
    177 004120      042     103     110  SER16:  .ASCII  \"CHECKSUM ERROR ON LEVEL 0 RECEIVE'N\
    178 004142      000                          .BYTE   0
    179 004143      042     122     105  SER17:  .ASCII  \"RESPONSE LONGER THAN EXPECTED FOR CMD'N\
    180 004167      000                          .BYTE   0
    181 004170      042     104     122  SER18:  .ASCII  \"DRIVE "R1\
    182 004175      000                          .BYTE   0
    183 004176      104     061     062  SER18D: .ASCII  \D12",  "\       ; NOTE: THE STRING WITHIN THE \\'S <<MUST>>
    184 004202      104     061     062  SER18C: .ASCII  \D12",  "\       ; BE AN EVEN NUMBER OF CHAR, BECAUSE THE
    185 004206      104     061     062  SER18B: .ASCII  \D12",  "\       ; LABELS WILL FORCE WORD ALINGMENT, LEAVING
    186 004212      104     061     062  SER18A: .ASCII  \D12N\           ; JUNK IN THE LAST BYTE. (SER18A OK TO BE ODD)
    187 004214      000                          .BYTE   0
    188 004215      042     104     122  SER40:  .ASCII  \"DRIVE NOT AVAILABLE TO THIS UDA'N\
    189 004236      000                          .BYTE   0
    190 004237      042     125     116  SER41:  .ASCII  \"UNSPINABLE DRIVE "R1\
    191 004251      000                          .BYTE   0
    192 004252      042     122     105  STAT0:  .ASCII\"REAL TIME STATE"S2H16N\
    193 004265      042     123     124          .ASCII\"STATUS (R TO L)"S2H16S2H16S2H16S2H16S2H16S2H16S2H16N\
    194 004320      000                          .BYTE   0
    195 004320                                   DMEND
        004321  013364                           .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
    196         000001                           .END
```

UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42 PAGE 60-4
SYMBOL TABLE

```
ARGS$ = 000003
ATTN  = 000002
AVAIL = 000100
BF.DAT= 000000
BF.ECC= 000401
BF.EDC= 000400
BREAK = 000000
BUFFLG= 040000
BUFSIZ= 000043
BUF1    001221
CHECK = 000010
CHGMOD= 000201
CHRRES= 000170
CLRBUF  004677
CLRDRV  004211
COMPAR= 000006
COMPLT= 000176
CONMEM  004316
CR.DIA  001454
CR.DRC  001176
CR.GCR  001437
CR.GST  001170
CR.INR  001472
CR.ONL  001445
CR.RDM  001421
CR.RUN  001464
CR.WRM  001205
CVT   = 000020
DATAVL= 040000
DATPRE= 000005
DBNCYL= 000022
DIA     001461
DIAGDR  003101
DIAGNS  003054
DIAG.1  001144
DIA.EN= 000374
DIA.OP= 000003
DINIT = 000011
DISCON= 000204
DIV50   004544
DONE  = 060016
DONECD  002366
DO.DCL  003265
DO.DC4  003263
DO.DIX  003306
DO.DI0  003603
DO.DI1  003340
DO.DI2  003370
DO.DI3  003521
DO.DI7  003525
DO.DI8  003530
DRC     001203
DRCBYT  001204
DRCLR1  002503
DRTYPE= 000007
DRVCLR= 000005
DRVID = 000004
DRVONL= 000213
DRVRUN= 000014
DU.DFL= 020000
DU.FTL= 050000
DU.INF= 030000
DU.QUE= 010000
DU.SPC= 060000
DU.TER= 040000
D.LIMT= 000001
D.SCHR= 000002
ECC   = 000015
ECCFLG= 010000
ECCRSH= 000002
ECHO  = 000010
ECHOC = 000350
ECHOD   001162
ECHO1   002430
ECHO1A  002432
ECHO2   002433
ECHO3   002452
ECHO4   002457
ECHO5   002473
EOC   = 100000
ERECOV= 000006
ERHARD= 100000
ERLEV = 000002
ERRMC = 060014
ERRMES= 060013
ERRN  = 004000
ERRTYP= 100000
ERSOFT= 140000
ER.END  004204
ERS000  003653
EXIT  = 000021
EX.END  004210
FB.DAT= 000000
FB.EDC= 000400
FCTSIZ= 000010
FFFC  = 177774
FFFD  = 177775
FFFE  = 177776
FNDASC  004560
FORMAT= 000001
FRAME = 000004
FREE    005077
FSTOP = 100000
FTLDEV= 040000
FTLSYS= 000000
FT.BUF= 000000
FT.HI = 000001
FT.LOW= 000002
GCC.EN= 000170
GCR     001444
GETBCN  004356
GETCHR= 000207
GETST   001175
GETSTA= 000011
GETSUB= 000210
GRPCYL= 000002
GRPOFF= 000011
GSTS3   002504
GSTS6   002505
GSTS7   002574
GTSTAT  004131
GT.CMD  002725
HBHINB= 007777
HBLONB= 170377
HDRPRE= 000005
HD.BAD= 110000
HD.DBN= 140000
HD.LBN= 000000
HD.PRV= 050000
HD.RBN= 060000
HD.REV= 030000
HD.XBN= 120000
HEADER= 000002
HIBYTE= 177400
HICYL = 000001
HIDBN = 000003
HILBN = 000002
HIMEM = 007774
HIRBN = 000003
HIXBN = 000002
HOSTRQ  004643
HRDREV= 000003
ILOOP   002377
INR     001477
INSEEK= 000012
IN.RQ   001543
IN.01   001544
IN.02   001545
IN.03   001546
IN.04   001547
IN.05   001550
IN.06   001551
IN.07   001552
IN.08   001553
IN.09   001554
IN.10   001555
IN.11   001556
IN.12   001557
IN.13   001560
IN.14   001561
IN.15   001562
IN.16   001563
IN.17   001564
IN.18   001565
IN.19   001566
IN.20   001567
IN.21   001570
IN.22   001571
IN.23   001572
IN.24   001573
IN.25   001574
IN.26   001575
IN.27   001576
IN.28   001577
IN.29   001600
IN.30   001601
IN.31   001602
IN.32   001603
IN.33   001604
IN.34   001605
IRECLB= 000216
LARGE = 000001
LBHINB= 177417
LBLONB= 177760
LBNCYL= 000000
LBNHST= 000012
LBNTRK= 000011
LINKLN= 000007
LOBYTE= 000377
LONG    001454
LONGTO= 000001
LOW   = 000002
LUNIT   001152
MAXSND= 001750
MEDTYP= 000006
MESSAG= 060015
MICREV= 000003
MRD   = 000016
MSG1    003242
MSG2    003342
MSR0    003374
MSR1    003423
MSR2    003453
MSR3    003514
MSR4    003555
MSR5    003612
MS2000  000000
MS2001  000043
MS2002  000126
MS2003  000165
MS2004  000235
MS2005  000275
MS2006  000340
MS2007  000421
MS2008  000500
MS2009  000530
MS2010  000564
MS2011  000607
MS2012  000674
MS2013  000735
MS2014  001001
MS2015  001035
MS2016  001133
MS2017  001224
MS2018  001255
MS2019  001312
MS2020  001336
MS2021  001424
MS2022  001511
MS2023  001554
MS2024  001621
MS2025  001653
MS2026  001712
MS2027  001737
MS2028  002027
MS2029  002062
MS2030  002121
MS2031  002147
MS2032  002237
MS2033  002265
MS2034  002317
MS2035  002340
MS2036  002424
MS2037  002456
MS2038  002514
MS2039  002541
MS2040  002631
MS2041  002663
MS2042  002721
MS2043  002746
MS2044  003035
MS2045  003067
MS2046  003125
MS2047  003152
MWR   = 000017
NAM.1   001145
NAM.2   001146
NAM.3   001147
NUMPTR= 000003
OK.END  004207
ONL     001452
OUT.RQ  001500
OUT.01  001501
OUT.02  001502
OUT.03  001503
OUT.04  001504
OUT.05  001505
OUT.06  001506
OUT.07  001507
OUT.08  001510
OUT.09  001511
OUT.10  001512
OUT.11  001513
OUT.12  001514
OUT.13  001515
OUT.14  001516
OUT.15  001517
OUT.16  001520
OUT.17  001521
OUT.18  001522
OUT.19  001523
OUT.20  001524
OUT.21  001525
OUT.22  001526
OUT.23  001527
```

UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42 PAGE 60-5
SYMBOL TABLE

```
OUT.24  001530          RDM     001426          SENDHR  001151          ST.DIA  002664          T2DLL = 060001
OUT.25  001531          RDMEM   003633          SER10   003772          ST.DR = 000040          T2STRT  002230
OUT.26  001532          RDM.EN= 000162          SER11   004005          ST.ERR= 000002          T4BB1 = 060005
OUT.27  001533          RDM.OP= 000215          SER12   004023          ST.FE = 000200          T4BB2 = 060006
OUT.28  001534          RDSTAT  004612          SER13   004035          ST.FO = 002000          T4MPRM= 060003
OUT.29  001535          READ    003020          SER14   004050          ST.MOD= 000001          T4MXFR= 060011
OUT.30  001536          RECAL   004055          SER15   004076          ST.MSK= 000000          T4SEEK= 060010
OUT.31  001537          REGS$ = 177777          SER16   004120          ST.OA = 000200          T4SOFT= 060007
OUT.32  001540          REGU$ = 000001          SER17   004143          ST.PE = 000040          T4UPRM= 060004
OUT.33  001541          RERRCA  005063          SER18   004170          ST.PS = 000002          UDADM2= 001000 G
OUT.34  001542          RERROR  005016          SER18A  004212          ST.RE = 000100          UNITNB  000740
OVERFL= 000002          RERRPA  005057          SER18B  004206          ST.RR = 000100          UNITS   000720
OVE.MN= 000714          RETS  = 000001          SER18C  004202          ST.RTY= 000003          UNIT0 = 000001
OVE.MS= 000000          REVECT= 000004          SER18D  004176          ST.RU = 000001          UNIT1 = 000002
OVL.MN= 004164          REVS  = 000007          SER18E  001156          ST.SR = 000020          UNIT2 = 000004
OVL.MS= 004322          RM    = 000004          SER40   004215          ST.STA= 000001          UNIT3 = 000010
OVL...= 010506          RREAL = 013400          SER41   004237          ST.S7 = 000400          UNSSUC= 000175
OVSTRT= 007774          RSTOP = 100000          SHRTTO= 000000          ST.UNT= 000000          UREAD = 000013
OVS.MN= 001040          RUN     001471          SNDAGN  004650          ST.WE = 000010          UTOTST= 060012
OVS.MS= 011410          RUNDR   004001          SPADJU  005074          SUB   = 000013          UWRITE= 000014
OV... = 011126          RWRDY = 100000          SS    = 000001          TALKER  004706          WAITSI= 000012
PHYSA = 001000          RW.ANG= 000006          ST      000744          TALK1A  004720          WBUFLN= 000401
PORT0   002314          RW.BUF= 000001          STACK   001646          TALK1B  004730          WCONT = 040000
PORT2   002315          RW.CMD= 000004          STARI   001647          TALK2A  004765          WREAL = 122400
PORT3   002345          RW.HI = 000003          STATEX  004640          TALK2B  004766          WRITE   002754
PORT4   002353          RW.LOW= 000002          STATLP  004616          TEST    002372          WRM     001212
PORT5   002362          RW.SDI= 000005          STATOK  004631          TESTEX  003100          WRM.OP= 000017
PRMS  = 000002          RW.STA= 000000          STATUS= 000007          TESTX   002356          WRONG = 000002
PTYPE$= 000020          SAVREG  001153          STAT0   004252          TIMEOU= 000001          WRTMEM  003721
RBNTRK= 000004          SAVRID  000743          STOSTA  004265          TO      005001          WSTOP = 140000
RBUFLN= 000415          SAVSTA  000742          STSIZE= 000200          TOOBIG= 000001          XBNCYL= 000021
RCONT = 000000          SBCRES= 000167          STSRES= 000366          TRKGRP= 000003          XFERRI= 000000
RCTCPS= 000001          SDI     000741          ST.C  = 000002          TYPERR  004504          XMTERR= 000400
RCTCSZ= 000014          SDILTO  001155          ST.CON= 000002          TOO     002646          XOR     004771
RCV   = 000005          SDISTO  001154          ST.DB = 001000          T1MSIZ= 060000          XREAD = 000002
RCVERR= 000004          SDIVER= 000000          ST.DF = 000020          T2CMD = 060002          XWRITE= 000003
RCVRDY= 000001          SEND  = 000004

. ABS.  022254      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  3845 WORDS  ( 16 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  70 PAGES
B:UDAT2.OBK,B:UDAT2/C=$DMACRO,B:UDAT2
```

UDAT2 DISK RESIDENT DMACR X04.01 13-AF R-82 18:06:42 PAGE S-1
CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

```
DO.DI8   40-24    41-6#
DO.DIX   33-4     39-19#
DONE      5-17#   26-41    28-43
DONECD   28-43#   28-45    29-27    29-29
DRC      20-15    20-16#
DRCBYT   20-17#
DRCLR1   31-10#   31-25
DRTYPE    4-15#
DRVCLR    5-35#   20-16
DRVID     4-14#
DRVONL    5-33#
DRVRUN    5-34#   20-44
DU.DFL    6-8#
DU.FTL    6-11#
DU.INF    6-9#
DU.QUE    6-7#
DU.SPC    5-3      5-4      5-5      5-6      5-7      5-8      5-9      5-10     5-11     5-12     5-13     5-14     5-15     5-16
          5-17     6-12#
DU.TER    6-10#
ECC       3-16#
ECCFLG    2-85#
ECCRSH    4-11#
ECHO      3-11#   30-14
ECHO1    29-34    30-11#
ECHO1A   30-12#   30-23
ECHO2    30-13#   30-40
ECHO3    30-21    30-26#
ECHO4    30-19    30-31#
ECHO5    30-25    30-27    30-32    30-37#
ECHOC     5-46#   20-2     20-3     20-4     20-5     20-6
ECHOD    20-2#    30-11
EOC       2-86#
ER.END   47-19    47-22    47-30    47-32#   48-19    48-22    48-30    48-32
ER5000   26-38    60-158#
ERECOV    5-31#
ERHARD    5-52#   27-41    29-26    29-28    29-37    30-24    30-26    30-33    31-27    31-38    31-41    31-48    31-50    32-10
          32-13    32-20    32-22    33-36    37-23    37-27    37-38    37-40    38-29    40-30    41-25    43-30    43-34    43-42
          43-44    44-26    44-30    44-38    44-40    45-9     45-12    45-19    45-21    46-9     46-12    46-19    46-21    47-18
          47-21    47-29    47-31    48-18    48-21    48-29    48-31
ERLEV     4-10#
ERRMC     5-15#   26-38
ERRMES    5-14#   59-39
ERRN      2-43#   27-41    27-41    27-41    27-41#   29-26    29-26    29-26    29-26#   29-28    29-28    29-28    29-28#   29-37
          29-37    29-37    29-37#   30-24    30-24    30-24    30-24#   30-26    30-26    30-26    30-26#   30-33    30-33    30-33
          30-33#   31-27    31-27    31-27    31-27#   31-38    31-38    31-38    31-38#   31-41    31-41    31-41    31-41#   31-48
          31-48    31-48    31-48#   31-50    31-50    31-50    31-50#   32-10    32-10    32-10    32-10#   32-13    32-13    32-13
          32-13#   32-20    32-20    32-20    32-20#   32-22    32-22    32-22    32-22#   33-36    33-36    33-36    33-36#   37-23
          37-23    37-23    37-23#   37-27    37-27    37-27    37-27#   37-38    37-38    37-38    37-38#   37-40    37-40    37-40
          37-40#   38-29    38-29    38-29    38-29#   40-30    40-30    40-30    40-30#   41-25    41-25    41-25    41-25#   43-30
          43-30    43-30    43-30#   43-34    43-34    43-34    43-34#   43-42    43-42    43-42    43-42#   43-44    43-44    43-44
          43-44#   44-26    44-26    44-26    44-26#   44-30    44-30    44-30    44-30#   44-38    44-38    44-38    44-38#   44-40
          44-40    44-40    44-40#   45-9     45-9     45-9     45-9#    45-12    45-12    45-12    45-12#   45-19    45-19    45-19
          45-19#   45-21    45-21    45-21    45-21#   46-9     46-9     46-9     46-9#    46-12    46-12    46-12    46-12#   46-19
          46-19    46-19    46-19#   46-21    46-21    46-21    46-21#   47-18    47-18    47-18    47-18#   47-21    47-21    47-21
          47-21#   47-29    47-29    47-29    47-29#   47-31    47-31    47-31    47-31#   48-18    48-18    48-18    48-18#   48-21
          48-21    48-21    48-21#   48-29    48-29    48-29    48-29#   48-31    48-31    48-31    48-31#
```

UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42 PAGE S-3
CROSS REFERENCE TABLE (CREF V04.00)

UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42 PAGE S-4
CROSS REFERENCE TABLE (CREF V04.00)

```
IN.02    21-46#   33-19*   34-11    35-10    37-13    40-77*   40-81*
IN.03    21-47#   34-12    35-11
IN.04    21-48#   34-13
IN.05    21-49#
IN.06    21-50#
IN.07    21-51#
IN.08    21-52#
IN.09    21-53#
IN.10    21-54#
IN.11    21-55#
IN.12    21-56#
IN.13    21-57#
IN.14    21-58#
IN.15    21-59#
IN.16    21-60#
IN.17    21-61#
IN.18    21-62#
IN.19    21-63#
IN.20    21-64#
IN.21    21-65#
IN.22    21-66#
IN.23    21-67#
IN.24    21-68#
IN.25    21-69#
IN.26    21-70#
IN.27    21-71#
IN.28    21-72#
IN.29    21-73#
IN.30    21-74#
IN.31    21-75#
IN.32    21-76#
IN.33    21-77#
IN.34    21-78#
IN.RQ    21-44#   21-79    55-20
INR      20-45    20-46#
INSEEK    5-40#
IRECLB    5-39#   20-46
LARGE     4-74#
LBHINB    4-58#
LBLONB    4-59#   24-24    28-28    32-25    32-29
LBNCYL    4-23#
LBNHST    4-39#
LBNTRK    4-37#
LINKLN    2-110#
LOBYTE    4-55#   34-15    40-44    40-48    50-32
LONG     20-37#   56-20
LONGTO    4-6#    32-28
LOW       4-70#
LUNIT    19-34#   27-48*   28-24*   34-24    35-16    38-33    40-6     42-5     51-53    55-14    56-28    59-19
MAXSND   19-45#   23-37
MEDTYP    4-35#
MESSAG    5-16#   38-33    51-53
MICREV    4-12#
MRD       3-17#   55-17
MS2000   27-41    60-6#
MS2001   29-26    60-9#
```

UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42 PAGE S-5
CROSS REFERENCE TABLE (CREF V04.00)

MS2002	29-28	60-12#				
MS2003	29-37	60-14#				
MS2004	30-24	60-16#				
MS2005	30-26	60-19#				
MS2006	30-33	60-22#				
MS2007	31-27	60-26#				
MS2008	31-38	60-29#				
MS2009	31-41	60-31#				
MS2010	31-48	60-33#				
MS2011	31-50	60-35#				
MS2012	32-10	60-39#				
MS2013	32-13	60-41#				
MS2014	32-20	60-43#				
MS2015	32-22	60-45#				
MS2016	33-36	60-49#				
MS2017	37-23	60-53#				
MS2018	37-27	60-55#				
MS2019	37-38	60-57#				
MS2020	37-40	60-59#				
MS2021	38-29	60-63#				
MS2022	40-30	60-69#				
MS2023	41-25	60-71#				
MS2024	43-30	60-73#				
MS2025	43-34	60-75#				
MS2026	43-42	60-77#				
MS2027	43-44	60-79#				
MS2028	44-26	60-83#				
MS2029	44-30	60-85#				
MS2030	44-38	60-87#				
MS2031	44-40	60-89#				
MS2032	45-9	60-93#				
MS2033	45-12	60-95#				
MS2034	45-19	60-97#				
MS2035	45-21	60-99#				
MS2036	46-9	60-103#				
MS2037	46-12	60-105#				
MS2038	46-19	60-107#				
MS2039	46-21	60-109#				
MS2040	47-18	60-113#				
MS2041	47-21	60-115#				
MS2042	47-29	60-117#				
MS2043	47-31	60-119#				
MS2044	48-18	60-123#				
MS2045	48-21	60-125#				
MS2046	48-29	60-127#				
MS2047	48-31	60-129#				
MSG1	38-33	60-133#				
MSG2	51-53	60-140#				
MSR0	52-22	60-142#				
MSR1	52-26	60-144#				
MSR2	52-30	60-146#				
MSR3	52-34	60-149#				
MSR4	52-38	60-152#				
MSR5	52-40	60-155#				
MWR	3-18#	55-22				
NAM.1	19-26#	41-18*	41-21*	41-25	41-25	

UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42 PAGE S-6
CROSS REFERENCE TABLE (CREF V04.00)

UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42 PAGE S-7
CROSS REFERENCE TABLE (CREF V04.00)

PORT$	28-18	28-20	28-22	28-40#										
PRM$	27-41	27-41	27-41	27-41	27-41	27-41	27-41	27-41	27-41	27-41	27-41#	29-26	29-26	29-26
	29-26	29-26	29-26	29-26	29-26	29-26	29-26	29-26#	29-28	29-28	29-28	29-28	29-28	29-28
	29-28	29-28	29-28	29-28	29-28#	29-37	29-37	29-37	29-37	29-37	29-37	29-37	29-37	29-37
	29-37	29-37#	30-24	30-24	30-24	30-24	30-24	30-24	30-24	30-24	30-24	30-24	30-24#	30-26
	30-26	30-26	30-26	30-26	30-26	30-26	30-26	30-26	30-26	30-26#	30-33	30-33	30-33	30-33
	30-33	30-33	30-33	30-33	30-33	30-33	30-33#	31-27	31-27	31-27	31-27	31-27	31-27	31-27
	31-27	31-27	31-27	31-27#	31-38	31-38	31-38	31-38	31-38	31-38	31-38	31-38	31-38	31-38
	31-38#	31-41	31-41	31-41	31-41	31-41	31-41	31-41	31-41	31-41	31-41	31-41#	31-48	31-48
	31-48	31-48	31-48	31-48	31-48	31-48	31-48	31-48	31-48#	31-50	31-50	31-50	31-50	31-50
	31-50	31-50	31-50	31-50	31-50	31-50#	32-10	32-10	32-10	32-10	32-10	32-10	32-10	32-10
	32-10	32-10	32-10#	32-13	32-13	32-13	32-13	32-13	32-13	32-13	32-13	32-13	32-13	32-13#
	32-20	32-20	32-20	32-20	32-20	32-20	32-20	32-20	32-20	32-20	32-20#	32-22	32-22	32-22
	32-22	32-22	32-22	32-22	32-22	32-22	32-22	32-22#	33-36	33-36	33-36	33-36	33-36	33-36
	33-36	33-36	33-36	33-36	33-36#	37-23	37-23	37-23	37-23	37-23	37-23	37-23	37-23	37-23
	37-23	37-23#	37-27	37-27	37-27	37-27	37-27	37-27	37-27	37-27	37-27	37-27	37-27#	37-38
	37-38	37-38	37-38	37-38	37-38	37-38	37-38	37-38	37-38	37-38#	37-40	37-40	37-40	37-40
	37-40	37-40	37-40	37-40	37-40	37-40	37-40#	38-29	38-29	38-29	38-29	38-29	38-29	38-29
	38-29	38-29	38-29	38-29#	40-30	40-30	40-30	40-30	40-30	40-30	40-30	40-30	40-30	40-30
	40-30#	41-25	41-25	41-25	41-25	41-25	41-25	41-25	41-25	41-25	41-25	41-25#	43-30	43-30
	43-30	43-30	43-30	43-30	43-30	43-30	43-30	43-30	43-30#	43-34	43-34	43-34	43-34	43-34
	43-34	43-34	43-34	43-34	43-34	43-34#	43-42	43-42	43-42	43-42	43-42	43-42	43-42	43-42
	43-42	43-42	43-42#	43-44	43-44	43-44	43-44	43-44	43-44	43-44	43-44	43-44	43-44	43-44#
	44-26	44-26	44-26	44-26	44-26	44-26	44-26	44-26	44-26	44-26	44-26#	44-30	44-30	44-30
	44-30	44-30	44-30	44-30	44-30	44-30	44-30	44-30#	44-38	44-38	44-38	44-38	44-38	44-38
	44-38	44-38	44-38	44-38	44-38#	44-40	44-40	44-40	44-40	44-40	44-40	44-40	44-40	44-40
	44-40	44-40#	45-9	45-9	45-9	45-9	45-9	45-9	45-9	45-9	45-9	45-9	45-9#	45-12
	45-12	45-12	45-12	45-12	45-12	45-12	45-12	45-12	45-12	45-12#	45-19	45-19	45-19	45-19
	45-19	45-19	45-19	45-19	45-19	45-19	45-19#	45-21	45-21	45-21	45-21	45-21	45-21	45-21
	45-21	45-21	45-21	45-21#	46-9	46-9	46-9	46-9	46-9	46-9	46-9	46-9	46-9	46-9
	46-9#	46-12	46-12	46-12	46-12	46-12	46-12	46-12	46-12	46-12	46-12	46-12#	46-19	46-19
	46-19	46-19	46-19	46-19	46-19	46-19	46-19	46-19	46-19#	46-21	46-21	46-21	46-21	46-21
	46-21	46-21	46-21	46-21	46-21	46-21#	47-18	47-18	47-18	47-18	47-18	47-18	47-18	47-18
	47-18	47-18	47-18#	47-21	47-21	47-21	47-21	47-21	47-21	47-21	47-21	47-21	47-21	47-21#
	47-29	47-29	47-29	47-29	47-29	47-29	47-29	47-29	47-29	47-29	47-29#	47-31	47-31	47-31
	47-31	47-31	47-31	47-31	47-31	47-31	47-31	47-31#	48-18	48-18	48-18	48-18	48-18	48-18
	48-18	48-18	48-18	48-18	48-18#	48-21	48-21	48-21	48-21	48-21	48-21	48-21	48-21	48-21
	48-21	48-21#	48-29	48-29	48-29	48-29	48-29	48-29	48-29	48-29	48-29	48-29	48-29#	48-31
	48-31	48-31	48-31	48-31	48-31	48-31	48-31	48-31	48-31	48-31#				
PTYPE$	27-41	27-41	27-41#	30-24	30-24	30-24#	30-26	30-26	30-26#	30-33	30-33	30-33	30-33	30-33#
	30-33#	31-41	31-41	31-41	31-41	31-41#	31-41#	31-50	31-50	31-50	31-50	31-50#	31-50#	32-13
	32-13	32-13	32-13	32-13#	32-13#	32-22	32-22	32-22	32-22	32-22#	32-22#	33-36	33-36	33-36#
	37-27	37-27	37-27	37-27	37-27#	37-27#	37-40	37-40	37-40	37-40	37-40#	37-40#	41-25	41-25
	41-25	41-25	41-25	41-25	41-25#	41-25#	41-25#	43-34	43-34	43-34	43-34	43-34#	43-34#	43-44
	43-44	43-44	43-44	43-44#	43-44#	44-30	44-30	44-30	44-30	44-30#	44-30#	44-40	44-40	44-40
	44-40	44-40#	44-40#	45-12	45-12	45-12	45-12	45-12#	45-12#	45-21	45-21	45-21	45-21	45-21#
	45-21#	46-12	46-12	46-12	46-12	46-12#	46-12#	46-21	46-21	46-21	46-21	46-21#	46-21#	47-21
	47-21	47-21	47-21	47-21#	47-21#	47-31	47-31	47-31	47-31	47-31#	47-31#	48-21	48-21	48-21
	48-21	48-21#	48-21#	48-31	48-31	48-31	48-31	48-31#	48-31#					
RBNTRK	4-31#													
RBUFLN	2-109#													
RCONT	2-81#													
RCTCPS	4-8#													
RCTCSZ	4-40#													
RCV	3-8#	23-57	56-35											
RCVERR	5-23#	54-17												

CROSS REFERENCE TABLE (CREF V04.00)

```
RCVRDY   5-21#  23-25   29-33
RDM     20-25   20-26#  35-10*  35-11*  35-12*  38-4*   38-5*   38-6*   38-18*  39-19*  39-20*  39-21*  51-82*
RDM.EN  19-51#  43-37   43-44   43-44
RDM.OP  19-48#  20-26
RDMEM   35-14   38-7    38-20   39-22   43-20#
RDSTAT  23-17   23-24   24-13   29-20   54-2#
READ    33-33   35-9#
RECAL   36-14   46-4#
REGS$   27-41   27-41   27-41#  29-26   29-26#  29-28   29-28#  29-37   29-37#  30-24   30-24   30-24#  30-26   30-26
        30-26#  30-33   30-33   30-33   30-33   30-33   30-33   30-33#  30-33#  30-33#  31-27   31-27#  31-38   31-38#
        31-41   31-41   31-41   31-41#  31-48   31-48#  31-50   31-50   31-50   31-50   31-50   31-50   31-50#  31-50#
        31-50#  32-10   32-10#  32-13   32-13   32-13   32-13#  32-20   32-20#  32-22   32-22   32-22   32-22   32-22
        32-22   32-22#  32-22#  32-22#  33-36   33-36   33-36   33-36   33-36   33-36#  33-36#  33-36#  37-23   37-23#
        37-27   37-27   37-27   37-27#  37-38   37-38#  37-40   37-40   37-40   37-40   37-40   37-40   37-40#  37-40#
        37-40#  38-29   38-29#  40-30   40-30#  41-25   41-25   41-25   41-25   41-25   41-25   41-25   41-25   41-25
        41-25#  41-25#  41-25#  43-30   43-30#  43-34   43-34   43-34   43-34#  43-42   43-42#  43-44   43-44   43-44
        43-44   43-44   43-44   43-44#  43-44#  43-44#  44-26   44-26#  44-30   44-30   44-30   44-30#  44-38   44-38#
        44-40   44-40   44-40   44-40   44-40   44-40   44-40#  44-40#  44-40#  45-9    45-9#   45-12   45-12   45-12
        45-12#  45-19   45-19#  45-21   45-21   45-21   45-21   45-21   45-21   45-21#  45-21#  45-21#  46-9    46-9#
        46-12   46-12   46-12   46-12#  46-19   46-19#  46-21   46-21   46-21   46-21   46-21   46-21   46-21#  46-21#
        46-21#  47-18   47-18#  47-21   47-21   47-21   47-21#  47-29   47-29#  47-31   47-31   47-31   47-31   47-31
        47-31   47-31#  47-31#  47-31#  48-18   48-18#  48-21   48-21   48-21   48-21#  48-29   48-29#  48-31   48-31
        48-31   48-31   48-31   48-31   48-31#  48-31#  48-31#
REGU$   30-33   30-33   30-33#  31-50   31-50   31-50#  32-22   32-22   32-22#  33-36   33-36   33-36#  37-40   37-40
        37-40#  41-25   41-25   41-25   41-25   41-25#  41-25#  41-25#  43-44   43-44   43-44#  44-40   44-40   44-40#
        45-21   45-21   45-21#  46-21   46-21   46-21#  47-31   47-31   47-31#  48-31   48-31   48-31#
RERRCA  59-33   59-38#
RERROR  27-41   29-26   29-28   29-37   30-24   30-26   30-33   31-27   31-38   31-41   31-48   31-50   32-10   32-13
        32-20   32-22   33-36   37-23   37-27   37-38   37-40   38-29   40-30   41-25   43-30   43-34   43-42   43-44
        44-26   44-30   44-38   44-40   45-9    45-12   45-19   45-21   46-9    46-12   46-19   46-21   47-18   47-21
        47-29   47-31   48-18   48-21   48-29   48-31   59-7#
RERRPA  59-34#  59-37
RETS     4-7#
REVECT   4-63#
REVS     4-16#
RM       4-32#
RREAL    2-84#
RSTOP    2-80#
RUN     20-43   20-44#
RUNDR   36-11   45-4#
RW.ANG   2-74#
RW.BUF   2-69#
RW.CMD   2-72#
RW.HI    2-71#
RW.LOW   2-70#
RW.SDI   2-73#
RW.STA   2-68#
RWRDY    5-26#
SAVREG  19-36#  27-29*  27-31*  27-32   30-33   30-33*  31-50   31-50*  32-22   32-22*  33-36   33-36*  37-40   37-40*
        40-46*  40-57   41-25   41-25*  43-44   43-44*  44-40   44-40*  45-21   45-21*  46-21   46-21*  47-31   47-31*
        48-31   48-31*
SAVRID  19-17#  39-30*  40-4    40-81
SAVSTA  19-16#  31-5*   31-24*  31-29*  33-2    36-2*   36-6*
SBCRES   5-44#
SDI     19-14#  27-49*  28-34*  29-13   36-5    36-10   36-15   37-15   38-10   38-23   39-4    39-26   40-53   42-4
        43-23   44-19   49-9
```

```
SDILTO  19-38#  32-31*  56-24
SDISTO  19-37#  32-4*   32-27*  56-22
SDIVER   4-4#
SEND     3-7#   23-43   56-15
SENDHR  19-31#  37-12*  37-17*  38-3*   56-26*
SER10   23-20   60-165#
SER11   23-29   60-167#
SER12   23-49   60-169#
SER13   23-65   60-171#
SER14   23-71   60-173#
SER15   23-75   60-175#
SER16   23-79   60-177#
SER17   23-81   60-179#
SER18   26-18   60-181#
SER18A  19-40   60-186#
SER18B  19-41   60-185#
SER18C  19-42   60-184#
SER18D  19-43   60-183#
SER18E  19-40#  26-26
SER40   23-34   60-188#
SER41   26-16   60-190#
SHRTTO   4-3#   32-24
SNDAGN  55-14#  55-19
SPADJU  59-32*  59-42#
SS       4-9#
ST      19-21#  20-12   20-15   20-19   20-25   20-30   20-33   20-39   20-43   20-45   24-8    31-20   31-21   32-24
        32-28   35-18   38-4    39-15   39-27   39-30   39-31*  39-33*  40-10   40-12   40-14   40-31   40-42   40-48*
        40-49*  42-12   49-13   49-16   50-28   51-47   51-56
ST.C     3-30#
ST.CON   3-29#
ST.DB    3-47#
ST.DF    3-44#
ST.DIA  33-2#
ST.DR    3-37#  33-3    39-16
ST.ERR   3-28#
ST.FE    3-41#  31-22
ST.FO    3-46#
ST.MOD   3-27#
ST.MSK   3-25#
ST.OA    3-35#
ST.PE    3-43#  31-22
ST.PS    3-39#
ST.RE    3-42#
ST.RR    3-36#
ST.RTY   3-31#
ST.RU    3-40#  36-2
ST.S7    3-48#
ST.SR    3-38#  36-6
ST.STA   3-26#
ST.UNT   3-24#
ST.WE    3-45#  31-22
STACK   18-8    22-6#   28-37
START   18-9    23-6#
STATO   49-7    60-192#
STATEX  54-16   54-18   54-22#
STATLP  54-9#   54-14   54-20
```

CROSS REFERENCE TABLE (CREF V04.00)

```
STATOK   54-12    54-17#
STATUS    3-10#   49-10    54-9
STOSTA   31-26    31-47    32-19    37-37    43-41    44-37    45-18    46-18    47-28    48-28    49-4#
STSIZE   19-20#   20-25    40-32    40-34
STSRES    5-45#   47-24    47-31    47-31
SUB       4-22#
T00      32-11    32-14    32-16    32-21    32-24#
T1MSIZ    5-3#
T2CMD     5-5#    33-24
T2DLL     5-4#    40-16
T2STRT   27-16#
T4BB1     5-8#
T4BB2     5-9#
T4MPRM    5-6#
T4MXFR    5-12#   56-31
T4SEEK    5-11#
T4SOFT    5-10#
T4UPRM    5-7#
TALK1A   56-17    56-20#
TALK1B   56-23    56-25#   56-43
TALK2A   56-37    56-46#
TALK2B   56-19    56-41    56-45    56-47#
TALKER   31-34    32-6     37-16    43-24    44-20    45-5     46-5     47-11    48-11    56-12#
TEST     28-35    29-11#
TESTEX   33-29    36-21#
TESTX    28-37#   36-21
TIMEOU    4-61#
TO       32-26    32-30    58-1#
TOOBIG    4-69#
TRKGRP    4-28#
TYPERR   31-40    32-12    37-26    43-33    44-29    45-11    46-11    47-20    48-20    52-16#
UDADM2    2-39#
UNIT0     4-46#   27-26    28-25
UNIT1     4-47#
UNIT2     4-48#
UNIT3     4-49#
UNITNB   19-11#   27-50*   28-23*   28-39    41-3*
UNITS    19-6#    23-14    25-7     25-19    25-26    26-7     26-36    27-24    28-13
UNSSUC    5-42#   31-45    32-17    37-35    43-39    44-35    45-16    46-16    47-26    48-26
UREAD     3-14#   40-35
UTOTST    5-13#   25-4
UWRITE    3-15#
WAITSI    3-13#
WBUFLN    2-108#   2-109
WCONT     2-79#
WREAL     2-83#
WRITE    33-31    34-10#
WRM      20-19    20-20#   33-6*    33-7*    33-8*    33-9*    33-10*   34-11*   34-12*   34-17*   34-18*   34-19*   40-4*    40-21*
         40-40*   40-41*   40-45*   40-50    40-58*   40-75*
WRM.OP   19-47#   20-20
WRONG     4-65#
WRTMEM   33-12    34-22    40-54    44-18#
WSTOP     2-78#
XBNCYL    4-41#
XFERRT    4-5#
XMTERR    5-25#   54-11
```

CROSS REFERENCE TABLE (CREF V04.00)

UDAT2 DISK RESIDENT DMACR X04.01 13-APR-82 18:06:42 PAGE M-1
CROSS REFERENCE TABLE (CREF V04.00)

UDAT3 DISK FUNCTIONAL DMACR X04.01 13-APR-82 18:08:21
TABLE OF CONTENTS

```
32-   1       STRST - STORE INFORMATION IN AREA 'ST'
33-   1       SEEK
34-   1       READ1 - READ SECTORS ON A TRACK FOR TEST
35-   1       UNITS, SDI COMMANDS AND PROGRAM VARIABLES
36-   1       DATA PATTERNS
37-   1       START, GETU<POLL ALL PORTS>, RBUFD, & FCHAIN
37-  20       GETU   - POLL ALL PORTS, THEN GET UNITS TO TEST
40-  48       REST OF FORMAT CHAIN
41-   1       MESSAGES
42-   1       OVERLAY 1  =  SETUP ( AND START TESTING )
42-   8       INIT DRIVE AND LOOK AT DRIVE SIGNALS
43-   1       GET DRIVE CHARACTERISTICS
43-  25       ISSUE ONLINE COMMAND
44-   1       ISSUE ONLINE COMMAND
44-  21       ISSUE DRIVE CLEAR COMMAND
45-   1       ISSUE CHANGE MODE COMMAND
45-  16       SPIN UP DRIVE
46-   1       ISSUE INITIATE RECALIBRATE COMMAND
47-   1       GET SUBUNIT CHARACTERISTICS
47-  25       SEEK TO CYLINDER 0, GROUP 0
48-   1       CALCULATE NUMBERS
49-   1       READ ALL FACTORY FORMATTED TRACKS
50-   1       SEEK TO SELECTED GROUP, READ HDR, SEEK TO DIAGNOSTIC AREA
51-   1       CHECK WRITE PROTECT
52-   1       CHANGE MODE TO ALLOW FORMATTING AND WRITING
52-  20       FORMAT A TRACK THEN CHECK IT.
53-   1       SEND INVALID LEVEL 2 COMMANDS
54-   1       SEND INVALID LEVEL 1 COMMANDS
55-   1       ISSUE DISCONNECT COMMAND
55-   9       CHECK AVAIL FLAG
56-   1       SNDLV1 AND TSTCMD
57-   1       FORTRK - FORMAT TRACK
58-   1       TRKTST - TEST TRACK
59-   1       WRTCMP - WRITE A DATA PATTERN AND COMPARE
60-   1       GENERATE A PATTERN
61-   1       WRITEB - WRITE A SECTOR
62-   1       READB - READ ONE SECTOR
63-   1       CMPDAT - COMPARE DATA
```

```
                                .TITLE  UDAT3   DISK FUNCTIONAL


;
;
; COPYRIGHT (C)  1981
; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.
;
;
; THIS  SOFTWARE  IS  FURNISHED  UNDER A LICENSE FOR USE ONLY ON A
; SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION
; OF  THE  ABOVE COPYRIGHT  NOTICE.  THIS SOFTWARE, OR  ANY  OTHER
; COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE  MADE AVAILABLE
; TO  ANY  OTHER PERSON  EXCEPT  FOR USE ON SUCH SYSTEM AND TO ONE
; WHO AGREES TO THESE LICENSE TERMS.  TITLE  TO  AND  OWNERSHIP OF
; THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DIGITAL.
;
; THE  INFORMATION  IN THIS DOCUMENT IS SUBJECT  TO CHANGE WITHOUT
; NOTICE AND SHOULD NOT BE CONSTRUED AS A  COMMITMENT  BY  DIGITAL
; EQUIPMENT CORPORATION.
;
; DIGITAL ASSUMES NO  RESPONSIBILITY FOR THE USE OR RELIABILITY OF
; ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
;
;
;
;
;
;
;
;THIS PROGRAM SHALL BE ASSEMBLED WITH THE PROGRAM DMACRO
;USING A COMMAND LINE SIMILAR TO:
;
;  UDAT3,UDAT3/C=[1,2]DMACRO,UDAT3
;
;THEN LINK THE OBJECT FILE USING A COMMAND LINE SIMILAR TO:
;
;  UDAT3.BIN/L=UDAT3
;
000000                  TEST4   =       0                       ; THIS IS NOT TEST4
```

```
UDAT3 DISK FUNCTIONAL DMACR X04.01 13-APR-82 18:08:21 PAGE 3
START OF TEST CODE

                                        .SBTTL  START OF TEST CODE
                                        
000000                                  DMCODE  UDADM3,0,714,3,0,1000
                                        
000714  114007                          CLR R0                          ;CHANGE TO BREAKPOINT FOR DEBUG
                                        
                                ;INITIALIZE STACK
                                        
000715  104206  001542                  MOV #STACK,SP                   ;SET UP STACK POINTER
000717  003652                          BR      START                   ; BRANCH OVER SUPPORT CODE
                                        
                                        
```

```
                                        .SBTTL  UDA DM PROGRAM PARAMETERS

                                        .LIST MEB
                                ;
                                ;       EQUATES
                                ;
                                ;       HIGHEST USABLE LOCATION OF UDA MEMORY + 1
                                ;
007774                          HIMEM   =       7774                    ; HIGH MEMORY+1
007774                          OVSTRT  =       7774                    ; OVERLAY ADDRESS LOCATION
                                ;
                                ;
                                ;       OFFSETS FOR FORMAT TRACK TABLE
                                ;
000000                          FT.BUF  =       0.                      ;BUFFER POINTER OFFSET
000001                          FT.HI   =       1.                      ;HI ORDER HEADER OFFSET
000002                          FT.LOW  =       2.                      ;LOW ORDER HEADER OFFSET
                                ;
                                ;
                                ;       OFFSETS FOR FORMAT TRACK BUFFER
                                ;
000000                          FB.DAT  =         0.                    ;FIRST DATA WORD OFFSET
000400                          FB.EDC  =       256.                    ;EDC WORD OFFSET
                                ;
                                ;
                                ;       OFFSETS FOR READ/WRITE I/O CHAIN TABLES
                                ;
000000                          RW.STAT =       0.                      ;STATUS AND NEXT BUFFER POINTER OFFSET
000001                          RW.BUF  =       1.                      ;POINTER TO DATA BUFFER
000002                          RW.LOW  =       2.                      ;HI ORDER EXPECTED HEADER
000003                          RW.HI   =       3.                      ;LOW ORDER EXPECTED HEADER
000004                          RW.CMD  =       4.                      ;SDI COMMAND AND HEAD ADDRESS
000005                          RW.SDI  =       5.                      ;DUMMY SDI CONTROL BLOCK POINTER
000006                          RW.ANG  =       6.                      ;THETA FROM INDEX
                                ;
                                ;       CONSTANTS FOR READ AND WRITE XFC'S
                                ;
140000                          WSTOP   =       140000                  ; LAST ENTRY IN CHAIN FOR WRITE
040000                          WCONT   =       40000                   ; WRITE CONTINUE
100000                          RSTOP   =       100000                  ; LAST ENTRY IN CHAIN FOR READ
000000                          RCONT   =       0                       ; READ CONTINUE
100000                          FSTOP   =       100000                  ; LAST ENTRY IN CHAIN FOR FORMAT
122400                          WREAL   =       122400                  ; WRITE REAL TIME ECOMMAND
013400                          RREAL   =       13400                   ; READ REAL TIME COMMAND
010000                          ECCFLG  =       10000                   ; ECC ERROR IN BUFFER BIT
100000                          EOC     =       100000                  ; END OF CHAIN
040000                          BUFFLG  =       40000                   ; BUFFER FULL OR EMPTY FLAG
001750                          MAXSND  =       1000.
                                ;
                                ;
                                ;       HEADER CODES
                                ;
000000                          HD.LBN  =       000000                  ;GOOD LBN
060000                          HD.RBN  =       060000                  ;GOOD RBN, PERHAPS UNUSED
030000                          HD.REV  =       030000                  ;REVECTORED LBN
110000                          HD.BAD  =       110000                  ;BAD BLOCK
050000                          HD.PRV  =       050000                  ;PRIMARY REVECTORED BLOCK
```

```
      58          120000                HD.XBN  =       120000                  ;XBN BLOCK
      59          140000                HD.DBN  =       140000                  ;DBN BLOCK
      60
      61                                ;       LEVEL 1 CODES
      62          070400                MS.STR  =       70400                   ;MESSAGE START
      63          131000                MS.END  =       131000                  ;MESSAGE END
      64          152000                MS.CNT  =       152000                  ;MESSAGE CONTINUE
      65          107000                SL.GRP  =       107000                  ;SELECT GROUP
      66
      67                                ;       OFFSETS FOR DATA BUFFERS
      68
      69          000000                BF.DAT  =         0.                    ;DATA
      70          000400                BF.EDC  =       256.                    ;ERROR DETECTION CODE
      71          000401                BF.ECC  =       257.                    ;LAST 17 ECC RESIDUES
      72
      73                                ;       BUFFER AND READ/WRITE CHAIN LINK SIZES
      74
      75          000401                WBUFLN  =       257.                    ; WRITE BUFFER SIZE
      76          000415                RBUFLN  =       WBUFLN+12.              ; READ BUFFER SIZE
      77          000007                LINKLN  =       7.                      ; LINK SIZE
      78
      79                                ;       UNIT PARAMETER USED BY TEST3
      80
      81          000050                U.SUBU  =       50
      82          000063                U.UNUM  =       63
      83
      84                                ;       FORMAT CHAIN SIZE
      85
      86          001375                FOR.SZ  =       255.*3
```

```
                                ;       XFC DEFINITION EQUATES
                                ;
000000                          BREAK   =       0.              ;BREAKPOINT XFC CODE
000001                          FORMAT  =       1.              ;FORMAT TRACK XFC CODE
000002                          XREAD   =       2.              ;READ N SECTORS XFC CODE
000003                          XWRITE  =       3.              ;WRITE N SECTORS XFC CODE
000004                          SEND    =       4.              ;SEND SDI COMMAND XFC CODE
000005                          RCV     =       5.              ;RECEIVE SDI MESSAGE XFC CODE
000006                          COMPARE =       6.              ;COMPARE DATA PATTERN TO BUFFER
000007                          STATUS  =       7.              ;RETURN DRIVE STATUS XFC CODE
000010                          ECHO    =       8.              ;ECHO DATA TO DRIVE XFC CODE
000011                          DINIT   =       9.              ;DRIVE INITIALIZE XFC CODE
000012                          WAITSI  =       10.             ;WAIT FOR SECTOR OR INDEX PULSE
000013                          UREAD   =       11.             ;READ UNIBUS MEMORY XFC CODE
000014                          UWRITE  =       12.             ;WRITE UNIBUS MEMORY XFC CODE
000015                          ECC     =       13.             ;DO ECC ON BUFFER XFC CODE
000016                          MRD     =       14.             ;SEND BUFFER TO MAINTENANCE READ COMMAND
000017                          MWR     =       15.             ;GET BUFFER FROM MAINTENANCE WRITE COMMAND
000020                          CVT     =       16.             ;CONVERT TO PHYSICAL ADDRESS XFC CODE
000021                          EXIT    =       17.             ;TERMINATE DM PROGRAM XFC CODE
                                ;
                                ;       GET STATUS OFFSETS
                                ;
000000                          ST.UNT  =       0.              ;UNIT NUMBER
000000                          ST.MSK  =       0.              ;SUBUNIT MASK
000001                          ST.STA  =       1.              ;STATUS BYTE
000001                          ST.MOD  =       1.              ;MODE BYTE
000002                          ST.ERR  =       2.              ;ERROR BYTE
000002                          ST.CON  =       2.              ;CONTROLLER BYTE
000002                          ST.C    =       2.              ;C BITS
000003                          ST.RTY  =       3.              ;RETRY COUNT/FAILURE CODE
                                ;
                                ;       STATUS BIT DEFINITIONS
                                ;
000010                          ST.EL   =       10              ; LOGGABLE INFO IN EXTENDED STATUS
000200                          ST.OA   =       200             ; ONLINE TO ANOTHER (SET IF DRIVE UNAVAILABLE)
000100                          ST.RR   =       100             ; READJUSTMENT BIT (SET IF RECALIBRATION REQUIRED)
000040                          ST.DR   =       40              ; DIAGNOSTIC REQUEST (SET IF DIAGNOSTIC REQUESTED)
000020                          ST.SR   =       20              ; SPINDLE READY (SET IF SPINDLE READY)
000002                          ST.PS   =       2               ; PORT SWITCH (SET IF PORT SWITCH IN)
000001                          ST.RU   =       1               ; RUN/STOP SWITCH (SET IF RUN/STOP SWITCH IN)
000200                          ST.FE   =       200             ; FATAL ERROR (SET IF FATAL ERROR OCCURRED)
000100                          ST.RE   =       100             ; RETRIABLE ERROR (SET IF RETRIABLE ERROR OCCURRED)
000040                          ST.PE   =       40              ; PROTOCOL ERROR (SET IF PROTOCOL ERROR OCCURRED)
000020                          ST.DF   =       20              ; INITIALIZATION FAILURE (SET IF INIT FAILED)
000010                          ST.WE   =       10              ; WRITE ENABLE (SET IF WRT ATTEMPTED ON PROT DISK)
002000                          ST.FO   =       2000            ; FORMATTING (SET IF FORMATTING ENABLED)
001000                          ST.DB   =       1000            ; DIAGNOSTIC CYLS (SET IF DIAG CYL ACCESS ENABLED)
000400                          ST.S7   =       400             ; SECTOR SIZE (SET FOR 576 BYTE SECTORS)
```

```
                                ;       GET COMMON CHARACTERISTICS OFFSETS
                                ;
000000          SHRTTO  =       0.              ;SHORT TIMEOUT <3:0>
000000          SDIVER  =       0.              ;SDI VERSION <7:4>
000000          XFERRT  =       0.              ;TRANSFER RATE <15:0>
000001          LONGTO  =       1.              ;LONG TIMEOUT <3:0>
000001          RETS    =       1.              ;RETRIES <7:4>
000001          RCTCPS  =       1.              ;F/RCT COPIES <11:8>
000001          SS      =       1.              ;SECTOR SIZE <15:15>
000002          ERLEV   =       2.              ;ERROR RETRY LEVELS <7:0>
000002          ECCRSH  =       2.              ;ECC THRESHOLD <15:8>
000003          MICREV  =       3.              ;MICROCODE REVISION NUMBER <7:0>
000003          HRDREV  =       3.              ;HARDWARE REVISION NUMBER <15:8>
000004          DRVID   =       4.              ;UNIQUE DRIVE ID <47:0>
000007          DRTYPE  =       7.              ;DRIVE TYPE IDENTIFIER <7:0>
000007          REVS    =       7.              ;REVS/SECOND <15:8>
                                ;
                                ;       GET SUBUNIT CHARACTERISTICS OFFSETS
                                ;
                                ;THESE OFFSETS ARE CURRENTLY GIVEN AS FOLLOWING THE COMMON CHARACTERISTICS
                                ;
000013          SUB     =       11.             ;OFFSET TO PUT SUBUNIT AFTER COMMON;
000000          LBNCYL  =       0.              ;NUMBER OF CYLINDERS IN LBN AREA <31:0>
000001          HICYL   =       1.              ;HI ORDER CYLINDER BITS <15:12>
000002          GRPCYL  =       2.              ;GROUPS PER CYLINDER <7:0>
000002          HILBN   =       2.              ;HI STARTING LBN <11:8>
000002          HIXBN   =       2.              ;HI STARTING XBN <15:12>
000003          TRKGRP  =       3.              ;TRACKS PER GROUP <7:0>
000003          HIRBN   =       3.              ;HI STARTING RBN <11:8>
000003          HIDBN   =       3.              ;HI STARTING DBN <15:12>
000004          RBNTRK  =       4.              ;RBNS PER TRACK <6:0>
000004          RM      =       4.              ;REMOVABLE MEDIA <7:7, 1=REMOVEABLE
000005          DATPRE  =       5.              ;DATA PREAMBLE SIZE IN WORDS <7:0>
000005          HDRPRE  =       5.              ;HEADER PREAMBLE SIZE IN WORDS <15:8>
000006          MEDTYP  =       6.              ;MEDIA TYPE <31:0>
000010          FCTSIZ  =       8.              ;FCT COPY SIZE <15:0>
000011          LBNTRK  =       9.              ;LBNS PER TRACK <7:0>
000011          GRPOFF  =       9.              ;GROUP OFFSET (SECTORS) <15:8>
000012          LBNHST  =       10.             ;LBNS IN HOST AREA <31:0>
000014          RCTCSZ  =       12.             ;RCT COPY SIZE <15:0>
000021          XBNCYL  =       17.             ;CYLS IN XBN AREA <15:0>
000022          DBNCYL  =       18.             ;CYLS IN DBN AREA <15:8>
                                ;
                                ;       UNIT CODES
                                ;
000001          UNIT0   =       1.              ;UNIT ZERO CODE
000002          UNIT1   =       2.              ;UNIT ONE CODE
000004          UNIT2   =       4.              ;UNIT TWO CODE
000010          UNIT3   =       8.              ;UNIT THREE CODE
                                ;
                                ;       BIT MASK DEFINITIONS
                                ;
177400          HIBYTE  =       177400          ;HIGH BYTE MASK
000377          LOBYTE  =       000377          ;LOW BYTE MASK
007777          HBHINB  =       7777            ;HI BYTE, HI NIBBLE MASK
170377          HBLONB  =       170377          ;HI BYTE, LO NIBBLE MASK
```

```
                177417          LBHINB  =       177417          ;LO BYTE, HI NIBBLE MASK
                177760          LBLONB  =       177760          ;LO BYTE, LO NIBBLE MASK
                                ;
                000001          TIMEOUT =       1.              ;DRIVE TIMEOUT CODE
                000002          HEADER  =       2.              ;HEADER COMPARE FAILURE CODE
                000004          REVECT  =       4.              ;REVECTOR NEEDED CODE
                                ;
                000002          WRONG   =       2.              ;FIRST WORD NOT START FRAME CODE
                000004          FRAME   =       4.              ;FRAMING ERROR CODE
                000010          CHECK   =       8.              ;CHECKSUM ERROR CODE
                                ;
                000001          TOOBIG  =       1.              ;NUMBER OF WORDS EXCEEDS 7064
                000002          LOW     =       2.              ;DM BUFFER ADDRESS IS LESS THAN 714
                                ;
                                ;       BIT MASK
                                ;
                000001          LARGE   =       1.              ;BLOCK NUMBER TOO LARGE
                000002          OVERFL  =       2.              ;SECTOR NUMBER LARGER THAN 16 BITS
```

```
                                ;MAINTANENCE READ/WRITE REQUEST NUMBERS
                                ;
060000          T1MSIZ  =       0.+60000        ;GET FREE MEMORY PARAMETERS
060001          T2DLL   =       1.+60000        ;DOWNLINE LOAD DRIVE DIAGNOSTIC
060002          T2CMD   =       2.+60000        ;MANUAL INTERVENTION TEST 2 PROTOCOL
060003          T4MPRM  =       3.+60000        ;GET MASTER PARAMETERS FROM SW QUESTIONS
060004          T4UPRM  =       4.+60000        ;GET UNIT PARAMETERS FROM HW QUESTIONS
060005          T4BB1   =       5.+60000        ;GET BAD BLOCKS (1 THRU 14)
060006          T4BB2   =       6.+60000        ;GET REST OF BAD BLOCKS (15 AND 16)
060007          T4SOFT  =       7.+60000        ;ADD TO SOFT ERROR AND ECC COUNT
060010          T4SEEK  =       8.+60000        ;ADD 1 TO SEEK COUNT
060011          T4MXFR  =       9.+60000        ;ADD TO MEGABITS READ AND WRITTEN
060012          UTOTST  =       10.+60000       ;GET UNITS TO TEST
060013          ERRMES  =       11.+60000       ;PRINT ERROR MESSAGE
060014          ERRMC   =       12.+60000       ;TEST 4 ERROR REPORTING
060015          MESSAG  =       13.+60000       ;INFORMATION MESSAGE
060016          DONE    =       14.+60000       ;MARK DM PROGRAM AS NO LONGER RUNNING
                                ;
                                ;       OTHER BIT DEFINITIIONS
                                ;
000001          RCVRDY  =       1               ; RECIEVER READY 1 = READY
000002          ATTN    =       2               ; ATTENTION BIT FOR RETURN DRIVE SIGNALS XFC
000004          DCLOCK  =       4               ; TRANSMIT ERROR
000100          AVAIL   =       100             ; AVAILABLE 1 = AVAILABLE
000400          RCVERR  =       400             ; RECIEVER ERROR
100000          RWRDY   =       100000          ; IF SET, UDA IS ABLE TO READ AND/OR WRITE TO DRIVE
                                ;
                                ;       SDI COMMANDS AND RESPONSES
                                ;
000204          DISCON  =       204             ; DISCONNECT DRIVE
000006          ERECOV  =       6               ; ERROR RECOVERY
000201          CHGMOD  =       201             ; CHANGE MODE
000213          DRVONL  =       213             ; DRIVE ONLINE
000014          DRVRUN  =       14              ; DRIVE RUN
000005          DRVCLR  =       5               ; DRIVE CLEAR OPCODE
000207          GETCHR  =       207             ; GET CHARACTERISTICS
000210          GETSUB  =       210             ; GET SUBUNIT CHARACTERISTICS
000011          GETSTA  =       11              ; GET STATUS
000216          IRECLB  =       216             ; RECALIBRATE
000012          INSEEK  =       12              ; INITIATE SEEK
000176          COMPLT  =       176             ; SUCCESSFUL COMPLETION
000175          UNSSUC  =       175             ; UNSUCCESSFUL COMPLETION
000170          CHRRES  =       170             ; GET CHARACTERISTICS RESPONSE
000167          SBCRES  =       167             ; GET SUBUNIT CHARACTERISTICS RESPONSE
000366          STSRES  =       366             ; GET STATUS RESPONSE
000350          ECHOC   =       350             ; DIAGNOSTIC ECHO COMMAND AND RESPONSE
                                ;
                                ;       ERROR CODES
                                ;
000000          FTLSYS  =          0            ; SYSTEM FATAL ERROR
040000          FTLDEV  =      40000            ; DEVICE FATAL
100000          ERHARD  =     100000            ; HARD ERROR
140000          ERSOFT  =     140000            ; SOFT ERROR
040000          C2HARD  =     <ERHARD&^CERSOFT>!<ERSOFT&^CERHARD>     ; CHANGE SOFT TO HARD ERROR
```

```
                                        .SBTTL  TEST 4 SPECIFIC INFORMATION
                                ;
                                ;       TEST 4 SPECIFIC INFORMATION
                                ;
                                ;
                                ;       CONSTANTS
                                ;
000377                  SCTWRD  =       255.                    ; NUMBER OF WORDS IN SECTOR TO FILL
000105                  INTEDC  =       69.                     ; INITIAL EDC VALUE
```

```
                        .SBTTL  MACRO DEFINITIONS
;
;       MESSAGE CONTROL TABLE MACRO
;
        .MACRO MSG CMDBUF,CMDSZ,RPLBUF,RPLSZ,SUCCOM
        .WORD CMDBUF                    ;ADDRESS OF COMMAND
        .WORD CMDSZ                     ;SIZE OF COMMAND IN BYTES
        .WORD RPLBUF                    ;ADDRESS OF REPLY
        .WORD RPLSZ                     ;SIZE OF REPLY IN WORDS
        .IF NB NUMBER
        .WORD SUCCOM                    ; SUCCESSFUL COMPLETION CODE
        .ENDC
        .ENDM
```

MACRO DEFINITIONS

```
      1                                       .MACRO  BCS     LAB..
      2                                                               BCC     .+2
      3                                                               BR      LAB..
      4                                       .ENDM
```

```
                                ;       PUSH REGISTER MACRO
                                ;
                                        .MACRO PUSH R9
                                        .IRP X,<R9>
                                                                        MOV X,-(SP)
                                        .ENDR
                                        .ENDM
                                ;
                                ;       POP REGISTER MACRO
                                ;
                                        .MACRO POP R9
                                        .IRP X,<R9>
                                                                        MOV (SP)+,X
                                        .ENDR
                                        .ENDM
```

```
                        ;ERROR MACROS
                        ;THESE MACROS ARE CALLED TO REPORT ERRORS TO THE HOST PROGRAM.
                        ;THE MACRO NAMES ARE : ERRSF, ERRDF, ERRHRD, ERRSFT. EACH RESULTS IN THE HOST
                        ;BEING REQUESTED TO REPORT THE ERROR.
                        ;ARGUMENTS:     1 (MS$) MESSAGE POINTER
                        ;               2 (P1$) PARAMETER #1
                        ;               3 (P2$) PARAMETER #2
                        ;               4 (P3$) PARAMETER #3
                        ;               5 (P4$) PARAMETER #4
                        ;               6 (P5$) PARAMETER #5
                        ;               7 (P6$) PARAMETER #6
                        ;               8 (P7$) PARAMETER #7
                        ;               9 (P8$) PARAMETER #8
                        ;
                        ;THE MESSAGE POINTER MUST POINT TO AN ADDRESS IN THE OVERLAY 'MS' IMMEDIATELY
                        ;FOLLOWING THE MAIN CODE. ANY ADDRESS MODE MAY BE USED (E.G. #MS1, @R2).
                        ;THE ADDRESS MUST CONTAIN AN ASCII FORMAT STRING TO DETERMINE THE MESSAGE
                        ;TO PRINT.
                        ;THE PARAMETER ARGUMENTS ARE OPTIONAL. THEY SHOULD BE SUPPLIED ONLY WHEN
                        ;THERE IS DATA TO BE PASSED TO THE HOST THAT WILL BE USED IN PRINTING THE
                        ;MESSAGE. THESE PARAMETER ARGUMENTS ARE THE ADDRESS OF DATA TO BE PASSED
                        ;USING ANY ADDRESSING MODE DESIRED.
                        ;ALL REGISTERS ARE RETURNED UNCHANGED. IT SHOULD BE NOTED THAT ARGUMENTS
                        ;CONTAINING SOMETHING OTHER THAN A REGISTER NAME (E.G. #100 OR MEMADR)
                        ;ASSEMBLE TO INSTRUCTIONS THAT SAVE AND RESTORE A REGISTER ON THE STACK.

                                .MACRO ERRSF MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$
                                .NARG ARGS$
                                .RADIX 10
                                ERROR$ FTLSYS,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,\ERRN
                                .ENDM

                                .MACRO ERRDF MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$
                                .NARG ARGS$
                                .RADIX 10
                                ERROR$ FTLDEV,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,\ERRN
                                .ENDM

                                .MACRO ERRHRD MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$
                                .NLIST
                                .NARG ARGS$
                                .RADIX 10
                                ERROR$ ERHARD,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,\ERRN
                                .LIST
                                .ENDM

                                .MACRO ERRSFT MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$
                                .NARG ARGS$
                                .RADIX 10
                                ERROR$ ERSOFT,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,\ERRN
                                .ENDM
```

```
                        ;THE FOLLOWING MACRO ACTUALLY PROCESSES THE ERROR CALL TO THE HOST PROGRAM

                            .MACRO ERROR$ ET$,MS$,P1$,P2$,P3$,P4$,P5$,P6$,P7$,P8$,ERRN$
                              .RADIX 8
                              PRMS=ARGS$-1
                              .IF LT,<PRMS>
                                .ERROR;NOT ENOUGH ARGUMENTS IN ERROR CALL
                              .ENDC
                              REGS$=-1
                              .IIF GE,<PRMS-8.>,PARGS. P8$
                              .IIF GE,<PRMS-7.>,PARGS. P7$
                              .IIF GE,<PRMS-6.>,PARGS. P6$
                              .IIF GE,<PRMS-5.>,PARGS. P5$
                              .IIF GE,<PRMS-4.>,PARGS. P4$
                              .IIF GE,<PRMS-3.>,PARGS. P3$
                              .IIF GE,<PRMS-2.>,PARGS. P2$
                              .IIF GE,<PRMS-1.>,PARGS. P1$
                              .IF GE REGS$
                                RSTR$ \REGS$
                              .ENDC
                              .RADIX 10
                              .LIST
                                                                          CALL RERROR     ;ERROR # ERRN$'.
                              .NLIST
                              .RADIX 8
                              .LIST
                                                                          .WORD ET$+ERRN
                                                                          .WORD <PRMS*10000>+MS$
                              .NLIST
                              ERRN=ERRN+1
                            .ENDM

                            .MACRO PARGS.,ADDR$
                              .NTYPE PTYPE$,ADDR$
                              .IF EQ,<PTYPE$&70>
                                .IIF EQ,<PTYPE$&7>-REGS$,RSTR$ \REGS$
                                .LIST
                                                                          MOV ADDR$,-(SP)
                                .NLIST
                              .IFF
                                .IF EQ,<PTYPE$&7>-1         ;PICK A REGISTER TO USE
                                  REGU$=2                   ;SELECT R2 IF R1 IS USED IN PARAMETER FETCH
                                .IFF
                                  REGU$=1                   ;OTHERWISE USE R1
                                .ENDC
                                .IF NE,<REGU$-REGS$>        ;IF REGISTER NOT ALREADY SAVED
                                  .IF GE,REGS$
                                    RSTR$ \REGS$            ;RESTORE CURRENT SAVED REGISTER
                                  .ENDC
                                  SAVR$ \REGU$              ;THEN SAVE SELECTED REGISTER
                                .ENDC
                                GETP$ \REGS$,ADDR$
                              .ENDC
                            .ENDM
```

MACRO DEFINITIONS

```
                                .MACRO SAVR$ REGN
                                  .LIST
                                                                MOV R'REGN,SAVREG
                                  .NLIST
                                  REGS$=REGN
                                .ENDM

                                .MACRO RSTR$ REGN
                                  .LIST
                                                                MOV SAVREG,R'REGN
                                  .NLIST
                                  REGS$=-1
                                .ENDM

                                .MACRO GETP$ REGN,ADDR$
                                  .LIST
                                                                MOV ADDR$,R'REGN
                                                                MOV R'REGN,-(SP)
                                  .NLIST
                                .ENDM
```

```
                        ;
                        ;       PRIMARY ERROR REPORTING (TEST 4)
                        ;

                        .MACRO  SOFTER  NUM,ARGS
                        ERROR   ERSOFT,NUM,<ARGS>
                                                                MOV     #ERRMES,OUT.RQ
                        .ENDM

                        .MACRO  REPSFT  SFTFLG,ECCFG,SEKFLG
                        .IF     NB,SFTFLG
                                                                MOV     #1,OUT.02
                        .IFF
                                                                CLR     OUT.02
                        .ENDC
                        .IF     NB,ECCFG
                                                                MOV     #1,OUT.03
                        .IFF
                                                                CLR     OUT.03
                        .ENDC
                        .IF     NB,SEKFLG
                                                                MOV     #1,OUT.04
                        .IFF
                                                                CLR     OUT.04
                        .ENDC
                                                                PUSH    R0
                                                                MOV     U.UNUM(R5),R0
                                                                ADD     U.SUBU(R5),R0
                                                                MOV     R0,OUT.01
                                                                MOV     #T4SOFT,R0
                                                                CALL    HOSTRQ
                                                                POP     R0
                        .ENDM

                        .MACRO  HARDER  NUM,ARGS
                        ERROR   ERHARD,NUM,<ARGS>
                                                                MOV     #ERRMES,OUT.RQ
                        .ENDM

                        .MACRO  DEVFTL  NUM,ARGS
                        ERROR   FTLDEV,NUM,<ARGS>
                                                                MOV     #ERRMES,OUT.RQ
                        .ENDM

                        .MACRO  SYSFTL  NUM,ARGS
                        ERROR   FTLSYS,NUM,<ARGS>
                                                                MOV     #ERRMES,OUT.RQ
                        .ENDM

                        .MACRO  ERROR   TYPE,NUM,ARGS
                        .RADIX  10
                        NUMPTR  =       5
                        MOVMSG  #MS'NUM,4
                        .IF     NB,<ARGS>
                        .IRP    X,<ARGS>
                        MOVMSG  X,\NUMPTR
                        NUMPTR  =       NUMPTR + 1
                        .ENDR
```

```
                                        .ENDC
                                                                        MOV     LUNIT,OUT.03
                                                                        MOV     #NUM!TYPE+3000.,R2
                                                                        MOV     R2,OUT.02
                                                                        MOV     #.,OUT.01
                                        .RADIX
                                        .ENDM

                                        .MACRO  CERROR  STNUM,ARGS
                                        .RADIX  10
                                NUMPTR  =       STNUM
                                        .IRP    X,<ARGS>
                                        MOVMSG  X,\NUMPTR
                                NUMPTR  =       NUMPTR + 1
                                        .ENDR
                                        .RADIX
                                        .ENDM

                                        .MACRO  ERRORC  ARGS
                                        .RADIX  10
                                        .IRP    X,<ARGS>
                                        MOVMSG  X,\NUMPTR
                                NUMPTR  =       NUMPTR + 1
                                        .ENDR
                                        .RADIX
                                        .ENDM

                                        .MACRO  MOVMSG  ARG,INDX

                                        .IF     LT,INDX-10
                                                                        MOV     ARG,OUT.0'INDX
                                        .IFF
                                                                        MOV     ARG,OUT.'INDX
                                        .ENDC
                                        .ENDM

                                        .MACRO  ENDERR  POS
                                        .RADIX  10
                                        .IF     NB,POS
                                        NDERR   POS
                                        .IFF
                                        NDERR   \NUMPTR
                                        .ENDC
                                        .RADIX
                                        .ENDM

                                        .MACRO  NDERR   POS
                                        .IF     NE,POS
                                        MOVMSG  #SER22,POS
                                                                        MOV     #POS+1,ERRPOS   ; SET THE POSITION
                                        .IFF
                                                                        CLR     ERRPOS          ; CLEAR THE POSITION
                                        .ENDC
                                        .ENDM
                                ;
                                ;       MESSAGE REPORTING MACRO
                                ;
```

```
                                .MACRO  MSSG    NUM,ARGS
                                .RADIX  10
                                NUMPTR  =       3
                                MOVMSG  #MS'NUM,2
                                .IF     NB,<ARGS>
                                .IRP    X,<ARGS>
                                MOVMSG  X,\NUMPTR
                                NUMPTR  =       NUMPTR + 1
                                .ENDR
                                .ENDC

                                                        PUSH    <R0,R1>
                                                        MOV     LUNIT,OUT.01
                                                        MOV     #MESSAG,R0
                                                        CALL    HOSTRQ
                                                        POP     <R1,R0>

                                .RADIX
                                .ENDM
                        ;
                        ;
                                .MACRO  MSSGE   NUM,ARGS
                                .RADIX  10
                                NUMPTR  =       3
                                MOVMSG  #'NUM,2
                                .IF     NB,<ARGS>
                                .IRP    X,<ARGS>
                                MOVMSG  X,\NUMPTR
                                NUMPTR  =       NUMPTR + 1
                                .ENDR
                                .ENDC

                                                        PUSH    <R0,R1>
                                                        MOV     #MESSAG,R0
                                                        CALL    HOSTRQ
                                                        POP     <R1,R0>

                                .RADIX
                                .ENDM
```

MACRO DEFINITIONS

```
      1                                 ;ASSUME MACRO
      2                                         .MACRO  ASSUME  P1,P2
      3                                         .IF     NE,P1-P2
      4                                         .ERROR  ;       THE ASSUMPTION THAT THE ABOVE VALUES ARE = IS FALSE
      5                                         .ENDC
      6                                         .ENDM
```

```
                        ;RETURN DRIVE STATUS MACRO WITH ERROR REPORTING

                                .MACRO  DSTAT,LAB$,E1,E2
                                .NLIST
                                .NLIST  MEB
                                .LIST   ME
                                .LIST
                                CALL    RDSTAT                  ; GET DRIVE STATUS
                                BIT     #10000,R1               ; SEE IF ANY ERRORS
                                BEQ     2$                      ; IF NO ERROR, BRANCH
                                BIT     #4000,R1                ; SEE IF XMIT ERROR
                                BEQ     1$                      ; IF SO, BRANCH
                                .NLIST  ME
                                .LIST   MEB
                                HARDER  E1                      ; REPORT INVALID STATUS ERROR
                                .NLIST  MEB
                                .LIST   ME
                                BR      LAB$                    ; BRANCH TO DONE
                        1$:
                                .NLIST  ME
                                .LIST   MEB
                                HARDER  E2                      ; REPORT XMIT ERROR
                                .NLIST  MEB
                                .LIST   ME
                                BR      LAB$                    ; BRANCH TO DONE
                        2$:
                                .NLIST
                                .NLIST  ME
                                .LIST   MEB
                                .LIST
                                .ENDM
```

MACRO DEFINITIONS

```
                        ;
                        ;       XOR THE CONTENTS OF TWO REGISTERS
                        ;
                        .MACRO  RXOR    REG1,REG2
                        MOV     REG2,-(SP)              ; SAVE REGISTER REG2
                        BIC     REG1,REG2               ; CLEAR COORESPONDING BITS IN REG2
                        BIC     (SP)+,REG1              ; CLEAR COORESPONDING BITS IN REG1
                        BIS     REG1,REG2               ; OR WHAT'S LEFT
                        .ENDM
```

```
                        ;
                        ;       CONVERT .BLKW CALLS TO ACTUAL WORDS GENERATED
                        ;
                        .MACRO  .BLKW   COUNT
                        .NLIST
                        .REPT   COUNT
                        .WORD   0
                        .ENDR
                        .LIST
                        .ENDM
```

```
                        ;       SDI INTERCHANGE WITH DRIVE WITH ERROR REPORTING

                                .MACRO  TALKX   ERRLAB,E1,E2
                                .NLIST
                                .NLIST  MEB
                                .LIST   ME
                                .LIST
                                CALL    TALK                    ; INITIATE SDI INTERCHANGE
                                TST     R2                      ; SEE IF ERROR OCCURRED
                                BEQ     12$                     ; IF NOT, BRANCH
                                BPL     11$                     ; IF SO, BRANCH
                                .NLIST  ME
                                .LIST   MEB
                                HARDER  E1                      ;SEND COMMAND ERROR
                                .NLIST  MEB
                                .LIST   ME
                                BR      ERRLAB
                        11$:
                                .NLIST  ME
                                .LIST   MEB
                                HARDER  E2                      ;RECEIVE COMMAND ERROR
                                .NLIST  MEB
                                .LIST   ME
                                BR      ERRLAB
                        12$:
                                .NLIST
                                .NLIST  ME
                                .LIST   MEB
                                .LIST
                                .ENDM
```

```
                                        .SBTTL  MACRO FOR OVERLAY TABLE
                                        .MACRO  DFOVLY  LABL$,ONAME,SAREA,EAREA
                                        .WORD   0
                                        .WORD   OVL.'ONAME'*4
                                        .IF     NE,OCNT$-LABL$
                                        .ERROR  ; OVERLAY NUMBER AND POSITION IN TABLE DO NOT MATCH
                                        .ENDC
                               OCNT$    =       OCNT$+1
                                        .ENDM
```

```
                                        .SBTTL  RTDS   - REAL TIME DRIVE STATE ROUTINE WITH ERROR REPORTING (TEST 4)
                                ;
                                ;       DIFF BETWEEN SEND + RECEIVE ERRORS, REPORT HERE
                                ;
000720  020746                 RTDS:   CALL    RTDSL           ; GET REAL TIME DRIVE STATE
000721  115002                         TST     R2              ; SEE IF ERROR OCCURRED
000722  010745                         BEQ     1$              ; IF NOT, BRANCH
000723                                 DEVFTL  13              ; REPORT DEVICE FATAL ERROR
000723  104200  000461  001037                                         MOV     #MS13,OUT.04
000726  104300  001475  001036                                         MOV     LUNIT,OUT.03
000731  104202  045705                                                 MOV     #13!FTLDEV+3000.,R2
000733  104020  001035                                                 MOV     R2,OUT.02
000735  104200  000735  001034                                         MOV     #.,OUT.01
000740  104200  060013  001033                                         MOV     #ERRMES,OUT.RQ
000743                                 ENDERR  0
000743  114000  003103                                                 CLR     ERRPOS          ; CLEAR THE POSITION
000745  000000                 1$:     RETURN
                                        .SBTTL  RTDSL - CALL RDSTAT
000746                         RTDSL:
000746  104302  002636                 MOV     SDI,R2                  ; GET INTERCONNECT CODE
000750  020754                         CALL    RDSTAT
000751  103201  077674                 BIC     #077674,R1              ; CLEAR UNUSED BITS
000753  000000                         RETURN
                                        .SBTTL  RDSTAT - GET DRIVE'S REAL TIME DRIVE STATE
000754                         RDSTAT:
                                ;
                                ;RETURN DRIVE STATUS
                                ;STATUS RETURNED IN DM REGISTER 1
                                ;
                                ;       INPUT R2 MUST HAVE INTERCONNECT CODE
                                ;
000754                                  PUSH    <R3,R0>                 ; SAVE R3 AND R0
000754  100463                                                                  MOV R3,-(SP)
000755  100467                                                                  MOV R0,-(SP)
000756  104203  024000                 MOV     #24000,R3               ; ERROR COUNT
000760  060007           1$:    XFC     STATUS                  ; GET DRIVE'S STATUS
000761  102201  000004                 BIT     #DCLOCK,R1              ; SE IF DRIVE CLOCK IS PRESENT
000763  010775                         BEQ     3$                      ; IF NOT, BRANCH
000764  102201  000400                 BIT     #RCVERR,R1              ; RECIEVER ERRORS
000766  010774                         BEQ     2$                      ; IF NOT VALID, BRANCH
000767  117403                         DEC     R3                      ; DECREMENT ERROR COUNT
000770  050760                         BNE     1$                      ; IF ERROR COUNT NON-ZERO, BRANCH
000771  104202  177777                 MOV     #177777,R2              ; MARK AS RECEIVE ERROR
000773  000775                         BR      3$
000774  114002           2$:    CLR     R2                      ; NO ERRORS
000775                   3$:    POP     <R0,R3>                 ; RESTORE R0, R3
000775  104267                                                                  MOV (SP)+,R0
000776  104263                                                                  MOV (SP)+,R3
000777  000000                         RETURN                          ; RETURN TO CALLING MODULE
```

```
                                        .SBTTL  HOSTRQ - HOST REQUEST - REPORT ERRORS, MEGABYTES TRANSFERRED, ETC.
001000                          HOSTRQ:
                                ;
                                ;SEND REQUEST BUFFER TO HOST AND WAIT FOR RESPONSE.
                                ;CLEAR ARGUMENT AREA OF OUT BUFFER IN PREPARATION
                                ;FOR NEXT HOSTRQ CALL.
                                ;
                                ;INPUTS:
                                ;       R0 - HOST REQUEST NUMBER
                                ;       OUT BUFFER LOADED WITH DATA

001000                          PUSH    <R0,R1,R2>
001000  100467                                                                  MOV R0,-(SP)
001001  100461                                                                  MOV R1,-(SP)
001002  100462                                                                  MOV R2,-(SP)
001003  104300  001475  001036          MOV     LUNIT,OUT.03
001006  104070  001033                  MOV     R0,OUT.RQ               ; STORE REQUEST NUMBER IN BUFFER
001010  104207  001033          SNDAGN: MOV     #OUT.RQ,R0              ; SEND BUFFER TO HOST
001012  104201  000043                  MOV     #BUFSIZ,R1
001014  060016                          XFC     MRD
001015  115001                          TST     R1                      ; CHECK FOR ERROR
001016  051010                          BNE     SNDAGN                  ; IF ERROR, TRY AGAIN
001017  104207  001033                  MOV     #OUT.RQ,R0              ; WAIT FOR RESPONSE FROM HOST
001021  104201  000043                  MOV     #BUFSIZ,R1
001023  060017                          XFC     MWR
001024  104200  177777  001033          MOV     #-1,OUT.RQ              ; MAKE REQUEST ILLEGAL
001027                                  POP     <R2,R1,R0>
001027  104262                                                                  MOV (SP)+,R2
001030  104261                                                                  MOV (SP)+,R1
001031  104267                                                                  MOV (SP)+,R0
001032  000000                          RETURN
```

```
                                ;STORAGE AREA FOR MAINTENANCE WRITE AND READ BUFFERS
                                ;
                                ;OUT BUFFER - DATA TO SEND TO HOST
                                ;
001033  000000  OUT.RQ: .WORD 0          ;HOST REQUEST CODE
001034  000000  OUT.01: .WORD 0          ;DATA ARGUMENT  1
001035  000000  OUT.02: .WORD 0          ;DATA ARGUMENT  2
001036  000000  OUT.03: .WORD 0          ;DATA ARGUMENT  3
001037  000000  OUT.04: .WORD 0          ;DATA ARGUMENT  4
001040  000000  OUT.05: .WORD 0          ;DATA ARGUMENT  5
001041  000000  OUT.06: .WORD 0          ;DATA ARGUMENT  6
001042  000000  OUT.07: .WORD 0          ;DATA ARGUMENT  7
001043  000000  OUT.08: .WORD 0          ;DATA ARGUMENT  8
001044  000000  OUT.09: .WORD 0          ;DATA ARGUMENT  9
001045  000000  OUT.10: .WORD 0          ;DATA ARGUMENT 10
001046  000000  OUT.11: .WORD 0          ;DATA ARGUMENT 11
001047  000000  OUT.12: .WORD 0          ;DATA ARGUMENT 12
001050  000000  OUT.13: .WORD 0          ;DATA ARGUMENT 13
001051  000000  OUT.14: .WORD 0          ;DATA ARGUMENT 14
001052  000000  OUT.15: .WORD 0          ;DATA ARGUMENT 15
001053  000000  OUT.16: .WORD 0          ;DATA ARGUMENT 16
001054  000000  OUT.17: .WORD 0          ;DATA ARGUMENT 17
001055  000000  OUT.18: .WORD 0          ;DATA ARGUMENT 18
001056  000000  OUT.19: .WORD 0          ;DATA ARGUMENT 19
001057  000000  OUT.20: .WORD 0          ;DATA ARGUMENT 20
001060  000000  OUT.21: .WORD 0          ;DATA ARGUMENT 21
001061  000000  OUT.22: .WORD 0          ;DATA ARGUMENT 22
001062  000000  OUT.23: .WORD 0          ;DATA ARGUMENT 23
001063  000000  OUT.24: .WORD 0          ;DATA ARGUMENT 24
001064  000000  OUT.25: .WORD 0          ;DATA ARGUMENT 25
001065  000000  OUT.26: .WORD 0          ;DATA ARGUMENT 26
001066  000000  OUT.27: .WORD 0          ;DATA ARGUMENT 27
001067  000000  OUT.28: .WORD 0          ;DATA ARGUMENT 28
001070  000000  OUT.29: .WORD 0          ;DATA ARGUMENT 29
001071  000000  OUT.30: .WORD 0          ;DATA ARGUMENT 30
001072  000000  OUT.31: .WORD 0          ;DATA ARGUMENT 31
001073  000000  OUT.32: .WORD 0          ;DATA ARGUMENT 32
001074  000000  OUT.33: .WORD 0          ;DATA ARGUMENT 33
001075  000000  OUT.34: .WORD 0          ;DATA ARGUMENT 34
        000043  BUFSIZ  =       . - OUT.RQ  ;SIZE OF BUFFER
```

```
                                        .SBTTL  TALK  - SDI LEVEL 2 INTERCHANGE ROUTINE
001076                          TALK:
                                ;
                                ;       TALK SENDS THE COMMAND TO THE DRIVE.  IF AN ERROR OCCURRS, R2 IS
                                ;       RETURNED NONZERO
                                ;
001076                                  PUSH    <R4,R5>         ; SAVE POINTER TO UNIT AND SUBUNIT PARAMETERS
001076  100464                                                                  MOV R4,-(SP)
001077  100465                                                                  MOV R5,-(SP)
001100  104302  002636                  MOV     SDI,R2          ; GET UNIT SDI SELECT MASK
001102  104205  000001                  MOV     #1,R5           ; SEND TIMEOUT DEFAULT IS ONE
001104  104207  002644                  MOV     #SNDONE,R0      ; MOVE SNDONE ADDRESS TO R0
001106  106037                          CMP     R3,R0           ; SEE IF ONLY TO BE SENT ONCE
001107  041112                          BCC     10$             ; IF SO, BRANCH
001110  104205  001750                  MOV     #MAXSND,R5      ; SEND MAXIMUM NUMBER OF TIMES (ONLINE)
001112  104137                  10$:    MOV     (R3),R0         ; POINTS TO SDI COMMAND BUFFER
001113  104631  000001                  MOV     1(R3),R1        ; LOAD BYTE COUNT
001115  060004                          XFC     SEND            ; SEND SDI COMMAND
001116  115001                          TST     R1              ; SEE IF SDI COMMAND SENT SUCESSFULLY
001117  011145                          BEQ     15$             ; IF SO, BRANCH
001120  117405                          DEC     R5              ; DECREMENT TIMEOUT
001121  051112                          BNE     10$             ; IF UNEXPIRED, BRANCH
001122                                  POP     R5              ; RESTORE R5
001122  104265                                                                  MOV (SP)+,R5
001123                                  PUSH    R3              ; SAVE SDI PACKET POINTER
001123  100463                                                                  MOV R3,-(SP)
001124                                  HARDER  1
001124  104200  000000  001037                                         MOV     #MS1,OUT.04
001127  104300  001475  001036                                         MOV     LUNIT,OUT.03
001132  104202  105671                                                 MOV     #1!ERHARD+3000.,R2
001134  104020  001035                                                 MOV     R2,OUT.02
001136  104200  001136  001034                                         MOV     #.,OUT.01
001141  104200  060013  001033                                         MOV     #ERRMES,OUT.RQ
001144  001335                          BR      50$             ; BRANCH
001145                          15$:    POP     R5              ; RESTORE R5
001145  104265                                                                  MOV (SP)+,R5
001146  106203  002707                  CMP     #LONG,R3        ; SEE IF LONG TIMEOUT TO BE USED
001150  071154                          BMI     20$             ; IF SO, BRANCH
001151  104304  001461                  MOV     SDISTO,R4       ; R4 HAS SHORT TIMEOUT
001153  001156                          BR      25$             ; BRANCH
001154  104304  001462          20$:    MOV     SDILTO,R4       ; R4 HAS LONG TIMEOUT
001156  104637  000002          25$:    MOV     2(R3),R0        ; POINT TO RECEIVE BUFFER
001160  104631  000003                  MOV     3(R3),R1        ; NUMBER OF WORDS IN RESPONSE
001162                                  PUSH    R3              ; SAVE POINTER TO COMMAND
001162  100463                                                                  MOV R3,-(SP)
001163  060005                          XFC     RCV             ; RECEIVE SDI RESPONSE
001164                                  POP     R3              ; RESTORE R3
001164  104263                                                                  MOV (SP)+,R3
001165  115001                          TST     R1              ; SEE IF SDI RESPONSE RECEIVED SUCESSFULLY
001166  011412                          BEQ     60$             ; IF SO, BRANCH
001167  106201  000001                  CMP     #1,R1           ; SEE IF TIMEOUT
001171  051216                          BNE     30$             ; IF NOT, BRANCH
001172  117404                          DEC     R4              ; DECREMENT TIMEOUT VALUE
001173  051156                          BNE     25$             ; IF TIMEOUT UNEXPIRED, BRANCH
001174                                  PUSH    R3              ; SAVE R3
001174  100463                                                                  MOV R3,-(SP)
001175                                  HARDER  2
```

```
                001175  104200  000014  001037                                          MOV     #MS2,OUT.04
                001200  104300  001475  001036                                          MOV     LUNIT,OUT.03
                001203  104202  105672                                                  MOV     #2!ERHARD+3000.,R2
                001205  104020  001035                                                  MOV     R2,OUT.02
                001207  104200  001207  001034                                          MOV     #.,OUT.01
                001212  104200  060013  001033                                          MOV     #ERRMES,OUT.RQ
     44         001215  001335                          BR      50$             ; BRANCH
     45         001216  106201  000002          30$:    CMP     #2,R1           ; SEE IF FIRST WORD NOT START FRAME
     46         001220  051242                          BNE     35$             ; IF NOT, BRANCH
     47         001221                                  HARDER  3
                001221  104200  000032  001037                                          MOV     #MS3,OUT.04
                001224  104300  001475  001036                                          MOV     LUNIT,OUT.03
                001227  104202  105673                                                  MOV     #3!ERHARD+3000.,R2
                001231  104020  001035                                                  MOV     R2,OUT.02
                001233  104200  001233  001034                                          MOV     #.,OUT.01
                001236  104200  060013  001033                                          MOV     #ERRMES,OUT.RQ
     48         001241  001335                          BR      50$             ; BRANCH
     49         001242  106201  000004          35$:    CMP     #4,R1           ; SEE IF FRAMING ERROR
     50         001244  051266                          BNE     40$             ; IF NOT, BRANCH
     51         001245                                  HARDER  4
                001245  104200  000063  001037                                          MOV     #MS4,OUT.04
                001250  104300  001475  001036                                          MOV     LUNIT,OUT.03
                001253  104202  105674                                                  MOV     #4!ERHARD+3000.,R2
                001255  104020  001035                                                  MOV     R2,OUT.02
                001257  104200  001257  001034                                          MOV     #.,OUT.01
                001262  104200  060013  001033                                          MOV     #ERRMES,OUT.RQ
     52         001265  001335                          BR      50$             ; BRANCH
     53         001266  106201  000010          40$:    CMP     #10,R1          ; SEE IF CHECKSUM ERROR
     54         001270  051312                          BNE     45$             ; IF NOT, BRANCH
     55         001271                                  HARDER  5
                001271  104200  000110  001037                                          MOV     #MS5,OUT.04
                001274  104300  001475  001036                                          MOV     LUNIT,OUT.03
                001277  104202  105675                                                  MOV     #5!ERHARD+3000.,R2
                001301  104020  001035                                                  MOV     R2,OUT.02
                001303  104200  001303  001034                                          MOV     #.,OUT.01
                001306  104200  060013  001033                                          MOV     #ERRMES,OUT.RQ
     56         001311  001335                          BR      50$             ; BRANCH
     57         001312  106201  000020          45$:    CMP     #20,R1          ; SEE IF BUFFER TOO SMALL
     58         001314  051353                          BNE     55$             ; IF NOT, BRANCH
     59         001315                                  HARDER  6
                001315  104200  000135  001037                                          MOV     #MS6,OUT.04
                001320  104300  001475  001036                                          MOV     LUNIT,OUT.03
                001323  104202  105676                                                  MOV     #6!ERHARD+3000.,R2
                001325  104020  001035                                                  MOV     R2,OUT.02
                001327  104200  001327  001034                                          MOV     #.,OUT.01
                001332  104200  060013  001033                                          MOV     #ERRMES,OUT.RQ
     60         001335                          50$:    CERROR  5,<#SER00,COMND>
                001335  104200  002347  001040                                          MOV     #SER00,OUT.05
                001340  104300  002637  001041                                          MOV     COMND,OUT.06
     61         001343                                  ENDERR  7               ; FLAG END OF REPORTING BUFFER
                001343  104200  003052  001042                                          MOV     #SER22,OUT.07
                001346  104200  000010  003103                                          MOV     #7+1,ERRPOS     ; SET THE POSITION
     62         001351                                  POP     R3              ; SAVE SDI PACKET POINTER
                001351  104263                                                                  MOV (SP)+,R3
     63         001352  001457                          BR      65$             ; EXIT
     64         001353                          55$:    HARDER  7               ; UNKNOWN ERROR RETURNED BY UDA
                001353  104200  000160  001037                                          MOV     #MS7,OUT.04
```

```
        001356  104300  001475  001036                                          MOV     LUNIT,OUT.03
        001361  104202  105677                                                  MOV     #7!ERHARD+3000.,R2
        001363  104020  001035                                                  MOV     R2,OUT.02
        001365  104200  001365  001034                                          MOV     #.,OUT.01
        001370  104200  060013  001033                                          MOV     #ERRMES,OUT.RQ
    65  001373                                  CERROR  5,<R1,#SER00,COMND>     ; REPORT ERROR
        001373  104010  001040                                                  MOV     R1,OUT.05
        001375  104200  002347  001041                                          MOV     #SER00,OUT.06
        001400  104300  002637  001042                                          MOV     COMND,OUT.07
    66  001403                                  ENDERR  8               ; FLAG END OF REPORTING BUFFER
        001403  104200  003052  001043                                          MOV     #SER22,OUT.08
        001406  104200  000011  003103                                          MOV     #8+1,ERRPOS     ; SET THE POSITION
    67  001411  001457                          BR      65$             ; EXIT
    68  001412  114002                  60$:    CLR     R2              ; FLAG AS NO ERROR OCCURED
    69  001413  106637  000004                  CMP     4(R3),R0        ; SEE IF COMMAND ACCEPTED
    70  001415  011457                          BEQ     65$             ; IF SO, BRANCH
    71  001416                                  HARDER  8
        001416  104200  000221  001037                                          MOV     #MS8,OUT.04
        001421  104300  001475  001036                                          MOV     LUNIT,OUT.03
        001424  104202  105700                                                  MOV     #8!ERHARD+3000.,R2
        001426  104020  001035                                                  MOV     R2,OUT.02
        001430  104200  001430  001034                                          MOV     #.,OUT.01
        001433  104200  060013  001033                                          MOV     #ERRMES,OUT.RQ
    72  001436                                  CERROR  5,<#SER00,COMND,4(R3),R0> ; REPORT FURTHER ERRORS
        001436  104200  002347  001040                                          MOV     #SER00,OUT.05
        001441  104300  002637  001041                                          MOV     COMND,OUT.06
        001444  104630  000004  001042                                          MOV     4(R3),OUT.07
        001447  104070  001043                                                  MOV     R0,OUT.08
    73  001451                                  ENDERR  9
        001451  104200  003052  001044                                          MOV     #SER22,OUT.09
        001454  104200  000012  003103                                          MOV     #9+1,ERRPOS     ; SET THE POSITION
    74  001457                          65$:    POP     R4              ; RESTORE R4
        001457  104264                                                          MOV (SP)+,R4
    75  001460  000000                          RETURN
    76
    77  001461  000012                  SDISTO: .WORD   10.             ; SDI SHORT TIMEOUT
    78  001462  000024                  SDILTO: .WORD   20.             ; SDI LONG TIMEOUT
```

```
                                                .SBTTL  TO - CALCULATE TIMEOUT INTERVALS
001463                                  TO:
                                        ;
                                        ;       CALCULATE THE TIMEOUT IN 9SEC INTERVALS (SDI RECEIVE XFC TAKES
                                        ;       9 SEC)
                                        ;
001463  104201  000001                          MOV     #1,R1           ; SET UP LOG2 SHIFTER
001465  105011                         1$:     ADD     R1,R1           ; DOUBLE THE TIMEOUT VALUE
001466  117407                                 DEC     R0              ; DECREMENT COUNT
001467  051465                                 BNE     1$              ; IF COUNT INCOMPLETE, BRANCH
001470  115407                         2$:     INC     R0              ; INCREMENT 9 SEC COUNT
001471  107201  000011                         SUB     #9.,R1          ; SUBTRACT 9 SEC FROM TIMEOUT
001473  031470                                 BPL     2$              ; IF MORE TIME TO GO, BRANCH
001474  000000                                 RETURN                  ; RETURN TO CALLING PROGRAM

001475  177777                  LUNIT:  .WORD -1                        ;LOGICAL UNIT NUMBER (-1 FOR NOT AVAILABLE)

                                                .SBTTL  LINIT - INIT THE DRIVE
001476  104302  002636          LINIT:  MOV     SDI,R2          ; R2 HAS INTERCONNECT CODE
001500  060011                          XFC     DINIT
001501  000000                          RETURN
```

```
                                        .SBTTL  STACK AREA
                                ;STACK AREA

001502  123456                          .WORD 123456                    ;END MARKER FOR STACK
001503                                  .BLKW 31.                       ;STACK
001542  123456                  STACK:  .WORD 123456                    ;MARKER FOR STACK UNDERFLOW

        002773                  SUB     =       SUB + CR                ;MODIFY SUB TO POINT AT SUBUNIT CHAR
        005670                          ERRN=3000.                      ;START ERROR NUMBERS AT 3000.

001543  003050                  SER18E: .WORD   SER18A                  ; FOR MULTIPLE SUBUNIT ERROR REPORTING
001544  003045                          .WORD   SER18B
001545  003042                          .WORD   SER18C
001546  003037                          .WORD   SER18D

                                ;
                                ;       EACH OVERLAY TABLE IS COMPOSED OF TWO WORDS:
                                ;
                                ;       0)      LOW ORDER STARTING ADDRESS OF OVERLAY
                                ;       1)      <1::0> HI ORDER STARTING UNIBUS ADDRESS
                                ;               <15::2> LENGTH OF OVERLAY
                                ;
        000000                  MSSG$   =       0
        000000                  OCNT$   =       0
001547                          OTABLE: DFOVLY  MSSG$,MS
001547  000000                          .WORD   0
001550  016234                          .WORD   OVL.MS*4
001551                                  DFOVLY  SETUP,SU
001551  000000                          .WORD   0
001552  003730                          .WORD   OVL.SU*4
001553                                  DFOVLY  TEST,TS
001553  000000                          .WORD   0
001554  011124                          .WORD   OVL.TS*4
        000003                  NUMOVL  =       <. - OTABLE> / 2
```

```
                                                .SBTTL  TEST 3 START AND LOOP ON UNITS
                                        ;         THE DIAGNOSTICS TO ALL UNITS SELECTED.
                                        ;TEST CODE WILL BE CALLED FOR EACH DISK SUBUNIT TO BE TESTED.
                                        ;  LUNIT WILL CONTAIN LOGICAL UNIT NUMBER OF DRIVE FOR ERROR REPORTS
                                        ;  SDI WILL CONTAIN SDI INTERCONNECT CODE FOR SELECTED DRIVE
                                        ;  UNITNB WILL CONTAIN AN EVEN NUMBER FOR TESTING FIRST SUBUNIT OF A DRIVE
                                        ;                      AN ODD NUMBER FOR TESTING SECOND SUBUNIT OF A DRIVE

001555  114001                  T3STRT: CLR     R1              ;START WITH UNIT 0 INDEX

001556  104207  002615          PORT2:  MOV     #UNITS,R0       ; GET POINTER TO UNITS TABLE
001560  105017                          ADD     R1,R0           ; ADD INDEX
001561  104173                          MOV     (R0),R3         ; GET CONTENTS OF TABLE
001562  031571                          BPL     1$              ; IF THIS UNIT IS PRESENT, BRANCH
001563  102201  000003                  BIT     #3,R1           ; SEE IF ON SUBUNIT 0 OF UNIT
001565  051623                          BNE     PORT5           ; IF NOT, TEST NEXT SUBUNIT
001566  105201  000003                  ADD     #3,R1           ; IF NO SUBUNIT 0, THEN NO UNIT - SKIP OTHER SUBUNITS
001570  001623                          BR      PORT5           ; BYPASS IF NO UNIT
001571  102203  040000          1$:     BIT     #40000,R3       ; SEE IF THIS UNIT IS TO BE TESTED
001573  051623                          BNE     PORT5           ; IF NOT, BRANCH
001574  104010  002635                  MOV     R1,UNITNB       ; STORE UNIT INDEX
001576  104030  001475                  MOV     R3,LUNIT        ; STORE LOGICAL UNIT NUMBER FOR DRIVE
001600  104202  000001                  MOV     #UNIT0,R2       ; GET UNIT 0 INTERCONNECT CODE
001602  110601                          ROR     R1              ; DIVIDE UNITNB BY FOUR
001603  110601                          ROR     R1
001604  103201  177760                  BIC     #LBLONB,R1      ; CLEAR UNUSED BITS
001606  117401                  PORT3:  DEC     R1
001607  071614                          BMI     PORT4           ; FOR EACH DRIVE OVER 0 SHIFT R2 LEFT
001610  110202                          ROL     R2
001611  103202  000001                  BIC     #1,R2           ; CLEAR CARRY ROTATED INTO REG (IF ANY)
001613  001606                          BR      PORT3
001614  104020  002636          PORT4:  MOV     R2,SDI          ; STORE SDI INTERCONNECT CODE
001616  001633                          BR      OVRLAY          ; BRING OVERLAY IN
                                                                ; PERFORM TEST ON THIS DRIVE
                                                                ; TEST RETURNS TO TESTX
001617  104206  001542          TESTX:  MOV     #STACK,SP       ; RESET STACK DUE TO JUMPS OUT OF
                                                                ; SUBROUTINES
001621  104301  002635                  MOV     UNITNB,R1       ; GET UNIT INDEX
001623  115401                  PORT5:  INC     R1              ; INCREMENT INDEX
001624  106201  000020                  CMP     #16.,R1         ; CHECK IF 16 DRIVES ALREADY SELECTED
001626  051556                          BNE     PORT2           ; REPEAT FOR ALL DRIVES
001627  104207  06J016          DONECD: MOV     #DONE,R0        ; END OF PROGRAM
001631  021000                          CALL    HOSTRQ
001632  001627                          BR      DONECD          ; REPEAT IF RETURNED
```

```
                                                .SBTTL  OVRLAY - OVERLAY PROCESS FOR BRINGING IN OVERLAY
                                        ; *** BRING IN OVERLAY
001633  104206  001542          OVRLAY: MOV     #STACK,SP               ; RESET STACK DUE TO JUMPS OUT OF
001635  104307  001730                  MOV     OVRLNM,R0               ; GET WHICH OVERLAY TO BRING IN
001637  103207  177776                  BIC     #177776,R0              ; CLEAR ALL BUT LAST BIT
001641  115407                          INC     R0                      ; INCREMENT POINTER
001642  104070  001731                  MOV     R0,OVRPNT               ; SAVE OVERLAY NUMBER FOR ERROR
001644  110207                          ROL     R0                      ; AND SHIFT FOR PROPER POINTER
001645  104205  000003                  MOV     #3,R5                   ; GET 3 TRIES
001647  104671  001550          1$:     MOV     OTABLE+1(R0),R1         ; GET THE HI ORDER BITS OF UNIBUS ADDRESS
001651  104012                          MOV     R1,R2                   ; GET NUMBER OF WORDS OF TRANSFER
001652  110602                          ROR     R2                      ; ROTATE BIS 0 CORRECT POSITION
001653  110602                          ROR     R2
001654  103202  140000                  BIC     #140000,R2              ; CLEAR UNUSED BITS
001656  103201  177774                  BIC     #177774,R1              ; CLEAR UNUSED BITS FOR HI ORDER
001660  104203  005247                  MOV     #OVRLPT,R3              ; POINT TO WHERE TO LOAD OVERLAY IN UDA RAM
001662  104677  001547                  MOV     OTABLE(R0),R0           ; GET LO ORDER UNIBUS ADDRESS
001664  060013                          XFC     UREAD                   ; READ THE OVERLAY INTO UDA MEMORY
001665  115001                          TST     R1                      ; DONE?
001666  011724                          BEQ     2$                      ; IF SO, BRANCH
001667  117405                          DEC     R5                      ; ELSE TRY AGAIN
001670  051647                          BNE     1$                      ; IF RETRIES NOT ALL USED, BRANCH
001671                                  DEVFTL  34,<OVRPNT,#SER39>              ; UNABLE TO READ FROM HOST MEMORY
001671  104200  002220  001037                      MOV     #MS34,OUT.04
001674  104300  001731  001040                      MOV     OVRPNT,OUT.05
001677  104200  002306  001041                      MOV     #SER39,OUT.06
001702  104300  001475  001036                      MOV     LUNIT,OUT.03
001705  104202  045732                              MOV     #34!FTLDEV+3000.,R2
001707  104020  001035                              MOV     R2,OUT.02
001711  104200  001711  001034                      MOV     #.,OUT.01
001714  104200  060013  001033                      MOV     #ERRMES,OUT.RQ
001717  104200  177777  001036              MOV     #-1,OUT.03              ; FLAG AS NOT ASSOCIATED WITH ANY UNIT
001722  021753                              CALL    TESTED                  ; AND REPORT ERROR
001723  001727                              BR      3$                      ; AND BRANCH
001724  115400  001730          2$:         INC     OVRLNM                  ; GET READY FOR NEXT OVERLAY
001726  025247                              CALL    OVRLPT                  ; GO EXECUTE CODE
001727  001617                  3$:         BR      TESTX                   ; HERE IF ERROR

001730  000000                  OVRLNM: .WORD   0                       ; OVERLAY NUMBER
001731  000000                  OVRPNT: .WORD   0
```

```
      1                                                .SBTTL  ERROR EXIT
      2 001732                                 TESTEW: CERROR  5,<#SER23,R1>          ; PRINT REAL TIME DRIVE STATE ONLY
        001732  104200  003120  001040                                                        MOV     #SER23,OUT.05
        001735  104010  001041                                                        MOV     R1,OUT.06
      3 001737  001753                                 BR      TESTED
      4 001740                                 TESTEV: ENDERR  5
        001740  104200  003052  001040                                                        MOV     #SER22,OUT.05
        001743  104200  000006  003103                                                        MOV     #5+1,ERRPOS     ; SET THE POSITION
      5 001746  104203  001033                 TESTEX: MOV     #OUT.RQ,R3             ; SET UP POSITION IN R3
      6 001750  105303  003033                         ADD     ERRPOS,R3
      7 001752  022131                         TESTEY: CALL    STRST                  ; GET STATUS FROM ST
      8 001753  104302  002636                 TESTED: MOV     SDI,R2                 ; R2 HAS INTERCONNECT CODE
      9 001755  060011                                 XFC     DINIT                  ; AND INIT DRIVE
     10 001756  104307  001033                         MOV     OUT.RQ,R0              ; PRINT ERROR
     11 001760  021000                                 CALL    HOSTRQ
     12 001761  104200  000377  002731  DR.CLR: MOV     #LOBYTE,ERRORS         ; CLEAR ALL ERRORS
     13 001764  104203  002645                         MOV     #CR.CLR,R3
     14 001766  021076                                 CALL    TALK
     15 001767  000000                                 RETURN                         ; GO TO CALLING ROUTINE
```

```
                                        .SBTTL  FNDCYL - FIND CYLINDER
                                ;
                                ;       FNDCYL  FIND CYLINDER
                                ;
                                ;       INPUTS: R4 -> HIGHEST CYL (VAR1)
                                ;               R5 -> DESIRED BN (VAR2)
                                ;               R3 = VAR3
                                ;               R1 = VAR4
                                ;
001770                          FNDCYL:
001770  106200  000104  003102        CMP     #'D,LETTER              ;IS IT A DIAGNOSTIC BLOCK?
001773  052006                        BNE     FND3
                                ; *** FOR DBN AREA
001774  104205  003047                MOV     #LCDBN,R5
001776  104204  003054        FNDCY2: MOV     #FDIACYL,R4
002000  103200  170000  003055        BIC     #^CHBHINB,FDIACYL+1
002003  104303  003060                MOV     SECTRK,R3
002005  002034                        BR      FND4
002006  106200  000114  003102  FND3:   CMP     #'L,LETTER              ;IS IT A LOGICAL BLOCK?
002011  052025                        BNE     1$                      ;IF NOT, BRANCH
                                ; *** FOR LBN AREA
002012  114000  003063                CLR     TSTCYL
002014  114000  003064                CLR     TSTCYL+1
002016  104204  003063                MOV     #TSTCYL,R4
002020  104303  003004                MOV     SUB+LBNTRK,R3           ; R3 = LBN'S PER TRACK (VAR3)
002022  103203  177400                BIC     #HIBYTE,R3
002024  002034                        BR      FND4                    ;CONTINUE
                                ; *** FOR XBN AREA
002025  104204  003056        1$:     MOV     #FXBNCYL,R4             ;SET POINTER TO FIRST XBN CYL->VAR2
002027  103200  170000  003057        BIC     #^CHBHINB,FXBNCYL+1
002032  104303  003060                MOV     SECTRK,R3               ;R3 = SECTORS PER TRACK (VAR3)
002034  104302  002774        FND4:   MOV     SUB+LBNCYL+1,R2
002036  103202  007777                BIC     #HBHINB,R2
002040  101642  000001                BIS     1(R4),R2
002042  100642  000001                MOV     R2,1(R4)
002044                          4$:
002044  104247                        MOV     (R4)+,R0                ;START LOADING VARIABLES
002045  104070  003110                MOV     R0,VAR1
002047  104147                        MOV     (R4),R0
002050  104070  003111                MOV     R0,VAR1+1
002052  104257                        MOV     (R5)+,R0
002053  104070  003112                MOV     R0,VAR2
002055  104157                        MOV     (R5),R0
002056  104070  003113                MOV     R0,VAR2+1
002060  104030  003114                MOV     R3,VAR3
002062  104010  003115                MOV     R1,VAR4
002064  104207  003110                MOV     #VAR1,R0                ; R0 -> VARIABLES
002066  104201  002773                MOV     #SUB,R1                 ; R1 -> SUBUNIT CHARACTERISTICS
002070  060020                        XFC     CVT                     ;CONVERT TO CYLINDER VALUE
002071  106200  000104  003102        CMP     #'D,LETTER              ; DON'T COMPARE IF IN DBN AREA
002074  012117                        BEQ     7$
002075  106300  003120  003107        CMP     GROUP,OLDGRP            ; HAVE WE GONE OVER 2 CYL BOUNDARIES?
002100  032114                        BPL     6$                      ; IF NOT, BRANCH
002101  107300  003070  003061        SUB     BLOCKC,TSTBLK           ; ELSE, GO BACK ONE CYLYINDER
002104  042107                        BCC     5$                      ; IF NOT CARRY, BRANCH
002105  117400  003062                DEC     TSTBLK+1
002107  114000  003107        5$:     CLR     OLDGRP                  ; NEW GROUP
```

```
     58 002111  117404                          DEC     R4                      ; RESET POINTERS
     59 002112  117405                          DEC     R5
     60 002113  001770                          BR      FNDCYL                  ; AND TRY AGAIN
     61
     62 002114  104300  003120  003107  6$:     MOV     GROUP,OLDGRP            ; SAVE IN OLD GROUP FOR NEXT TIME THROUGH
     63 002117  104300  003116  003063  7$:     MOV     CYLLO,TSTCYL            ;STORE IN TSTCYL
     64 002122  104300  003117  003064          MOV     CYLLO+1,TSTCYL+1
     65 002125  104300  003120  003065          MOV     GROUP,TSTCYL+2
     66 002130  000000                          RETURN
```

```
                                        .SBTTL  STRST - STORE INFORMATION IN AREA 'ST'
                                ;
                                ;       STRST   STORE ST
                                ;
                                ;       INPUT   R3 -> OUTPUT BUFFER
                                ;       OUTPUT  OUTPUT BUFFER FILLED WITH VALUES FROM ST
                                ;
 002131                         STRST:  PUSH    <R0,R1,R2>
 002131  100467                                                                  MOV R0,-(SP)
 002132  100461                                                                  MOV R1,-(SP)
 002133  100462                                                                  MOV R2,-(SP)
 002134  104302  002636                 MOV     SDI,R2                 ;R2 HAS INTERCONNECT CODE
 002136  060007                         XFC     STATUS                 ;GET DRIVE STATE
 002137  103201  077674                 BIC     #077674,R1             ;CLEAR UNUSED BITS
 002141  100231                         MOV     R1,(R3)+               ; STORE IN OUT BUFFER
 002142  104202  000007                 MOV     #7,R2                  ; R2 IS # OF WORDS TO MOVE
 002144  104207  002760                 MOV     #ST+7,R0               ; R0 -> STATUS
 002146  104471                 1$:     MOV     -(R0),R1
 002147  100231                         MOV     R1,(R3)+
 002150  117402                         DEC     R2
 002151  052146                         BNE     1$
 002152                                 POP     <R2,R1,R0>
 002152  104262                                                                  MOV (SP)+,R2
 002153  104261                                                                  MOV (SP)+,R1
 002154  104267                                                                  MOV (SP)+,R0
 002155  000000                         RETURN
```

```
                                        .SBTTL  SEEK
                                ;SEEK
                                ;
                                ;SEEK TO CYLINDER POINTED TO BY CONTENTS OF R1
                                ;
                                ;INPUTS:
                                ;       R1 - POINTER TO CYLINDER NUMBER
                                ;       R2 - SDI INTERCONNECT
                                ;
002156                          SEEK:   PUSH <R0,R1,R3>
002156  100467                                                                  MOV R0,-(SP)
002157  100461                                                                  MOV R1,-(SP)
002160  100463                                                                  MOV R3,-(SP)
002161  104302  002636                 MOV     SDI,R2
002163  104210  002741                 MOV (R1)+,INS+1                 ;PUT CYLINDER INTO COMMAND
002165  104210  002742                 MOV (R1)+,INS+2
002167  104110  002743                 MOV (R1),INS+3                  ;PUT GROUP INTO COMMAND
002171  104203  002703         SEEKA:  MOV #CR.SEK,R3                  ;POINT TO COMMAND
002173  104200  002462  002637         MOV     #MS.SEK,COMND           ; SET UP FOR ERROR
002176  021076                         CALL    TALK                    ; INITIATE SDI INTERCHANGE
002177  115002                         TST     R2                      ; SEE IF ERROR OCCURRED
002200  012203                         BEQ     SEEK1                   ; IF NOT, BRANCH
002201  021746                         CALL    TESTEX                  ; IF SO, REPORT
002202  002321                         BR      SEEK3                   ; AND EXIT
002203  114003                 SEEK1:  CLR R3                          ;SET UP WORST CASE SEEK TIME
002204  104302  002636         SEEK2:  MOV     SDI,R2                  ; MOVE MASK TO R2
002206  020754                         CALL    RDSTAT                  ; GET DRIVE STATUS
002207  115002                         TST     R2                      ; WAS IT OK?
002210  012260                         BEQ     2$                      ; IF SO, BRANCH
002211  102201  000400                 BIT     #RCVERR,R1              ; SEE WHICH ERROR
002213  052236                         BNE     1$                      ; AND BRANCH
002214                                 HARDER  10                      ; REPORT INVALID STATUS ERROR
002214  104200  000341  001037                 MOV     #MS10,OUT.04
002217  104300  001475  001036                 MOV     LUNIT,OUT.03
002222  104202  105702                         MOV     #10!ERHARD+3000.,R2
002224  104020  001035                         MOV     R2,OUT.02
002226  104200  002226  001034                 MOV     #.,OUT.01
002231  104200  060013  001033                 MOV     #ERRMES,OUT.RQ
002234  021732                         CALL    TESTEW                  ; BRANCH TO DONE
002235  002321                         BR      SEEK3
002236                          1$:
002236                                 HARDER  12                      ; REPORT XMIT ERROR
002236  104200  000434  001037                 MOV     #MS12,OUT.04
002241  104300  001475  001036                 MOV     LUNIT,OUT.03
002244  104202  105704                         MOV     #12!ERHARD+3000.,R2
002246  104020  001035                         MOV     R2,OUT.02
002250  104200  002250  001034                 MOV     #.,OUT.01
002253  104200  060013  001033                 MOV     #ERRMES,OUT.RQ
002256  021732                         CALL    TESTEW                  ; BRANCH TO DONE
002257  002321                         BR      SEEK3
002260                          2$:
002260  102201  100000                 BIT #RWRDY,R1                   ;IS R/W READY SET?
002262  052321                         BNE SEEK3                       ;  YES
002263  115403                         INC R3                          ;BUMP COUNT
002264  052204                         BNE SEEK2                       ;KEEP WAITING
002265                                 HARDER  27,<INS+1,INS+2,INS+3>  ;SEEK COMPLETE TIME OUT
002265  104200  001633  001037                 MOV     #MS27,OUT.04
```

```
                002270  104300  002741  001040                                  MOV     INS+1,OUT.05
                002273  104300  002742  001041                                  MOV     INS+2,OUT.06
                002276  104300  002743  001042                                  MOV     INS+3,OUT.07
                002301  104300  001475  001036                                  MOV     LUNIT,OUT.03
                002304  104202  105723                                          MOV     #27!ERHARD+3000.,R2
                002306  104020  001035                                          MOV     R2,OUT.02
                002310  104200  002310  001034                                  MOV     #.,OUT.01
                002313  104200  060013  001033                                  MOV     #ERRMES,OUT.RQ
     42         002316  104307  001033                  MOV     OUT.RQ,R0
     43         002320  021000                          CALL    HOSTRQ
     44         002321                          SEEK3:  POP <R3,R1,R0>
                002321  104263                                                          MOV (SP)+,R3
                002322  104261                                                          MOV (SP)+,R1
                002323  104267                                                          MOV (SP)+,R0
     45         002324  000000                  RETURN
```

```
                                        .SBTTL  READ1 - READ SECTORS ON A TRACK FOR TEST
                                ;READ1
                                ;
                                ;READ SECTORS ON THE SELECTED TRACK UNTIL ONE IS READ CORRECTLY
                                ;OR THE MAXIMUM NUMBER OF SECTORS IS READ
                                ;
                                ;INPUTS:
                                ;       R3 HAS MAX NUMBER OF SECTORS TO READ
                                ;       TRACK HAS TRACK NUMBER
                                ;       R5 -> POINTER TO BLOCK NUMBER OF FIRST SECTOR
                                ;OUTPUTS:
                                ;       R2 IS CLOBBERED
                                ;
002325                          READ1:  PUSH <R0,R1,R3,R4,R5>
002325  100467                                                                  MOV R0,-(SP)
002326  100461                                                                  MOV R1,-(SP)
002327  100463                                                                  MOV R3,-(SP)
002330  100464                                                                  MOV R4,-(SP)
002331  100465                                                                  MOV R5,-(SP)
002332  104304  003121                 MOV     TRACK,R4                ;R4 IS TRACK
002334  104207  003646                 MOV #RBUF0+RW.LOW,R0
002336  104251                          MOV (R5)+,R1                    ;PUT BLOCK NUMBER IN
002337  100271                          MOV R1,(R0)+                    ;  READ BUFFER
002340  104151                          MOV (R5),R1
                                ;
002341  106200  000130  003102         CMP     #'X,LETTER              ; DO WE DO PROCESS FOR XBN?
002344  052360                         BNE     9$                      ; IF NOT, BRANCH
                                ; *** FOR XBN AREA
002345  104305  002775                 MOV     SUB+HIXBN,R5            ; DO SPECIAL PROCESSING TO PUT HEADERIN PLACE
002347  110605                         ROR     R5                      ; SHIFT TO NEXT NIBBLE
002350  110605                         ROR     R5
002351  110605                         ROR     R5
002352  110605                         ROR     R5
002353  103205  170377                 BIC     #HBLONB,R5              ; STRIP OFF UNUSED PORTION
002355  101205  120000                 BIS     #HD.XBN,R5              ; SET HEADER CODE
002357  002403                         BR      10$
002360  106200  000104  003102 9$:     CMP     #'D,LETTER              ; DO WE PROCESS FOR DBN?
002363  012371                         BEQ     11$                     ; IF SO, BRANCH
                                ; *** FOR LBN AREA
002364  104305  002775                 MOV     SUB+HILBN,R5            ; GET HI LBN VALUE (DON'T HAVE TO ROR)
002366  103205  170377                 BIC     #HBLONB,R5
002370                                  ASSUME HD.LBN,0
002370  002403                         BR      10$
                                ; *** FOR DBN AREA
002371  104305  002776         11$:    MOV     SUB+HIDBN,R5            ; GET HIGH ORDER BITS OF STARTING DBN
002373  110605                         ROR     R5                      ; MOVE TO CORRECT POSITION
002374  110605                         ROR     R5
002375  110605                         ROR     R5
002376  110605                         ROR     R5
002377  103205  170377                 BIC     #HBLONB,R5              ; CLEAR UNUSED BITS
002401  101205  140000                 BIS     #HD.DBN,R5              ; SET HEADER CODE
002403  105051                 10$:    ADD     R5,R1
002404  100271                         MOV R1,(R0)+
002405  104171                         MOV (R0),R1                     ;PUT IN TRACK NUMBER
002406  103201  000377                 BIC #LOBYTE,R1                  ;(MERGE WITH REAL-TIME COMMAND)
002410  101041                         BIS R4,R1
002411  100171                         MOV R1,(R0)
```

```
     53 002412  104207  003644          READ1A: MOV #RBUF0,R0                   ;POINT TO READ BUFFER
     54 002414  104302  002636                  MOV     SDI,R2                   ; MOVE MASK TO R2
     55 002416  020754                          CALL    RDSTAT                   ; GET DRIVE STATUS
     56 002417  115002                          TST     R2                       ; WAS IT OK?
     57 002420  012470                          BEQ     2$                       ; IF SO, BRANCH
     58 002421  102201  000400                  BIT     #RCVERR,R1               ; SEE IF ANY ERRORS
     59 002423  052446                          BNE     1$                       ; REPORT
     60 002424                                  HARDER  10                       ; REPORT INVALID STATUS ERROR
        002424  104200  000341  001037                                                  MOV     #MS10,OUT.04
        002427  104300  001475  001036                                                  MOV     LUNIT,OUT.03
        002432  104202  105702                                                          MOV     #10!ERHARD+3000.,R2
        002434  104020  001035                                                          MOV     R2,OUT.02
        002436  104200  002436  001034                                                  MOV     #.,OUT.01
        002441  104200  060013  001033                                                  MOV     #ERRMES,OUT.RQ
     61 002444  021732                          CALL    TESTEW                   ; BRANCH TO DONE
     62 002445  002607                          BR      READ1X
     63 002446                          1$:
     64 002446                                  HARDER  12                       ; REPORT XMIT ERROR
        002446  104200  000434  001037                                                  MOV     #MS12,OUT.04
        002451  104300  001475  001036                                                  MOV     LUNIT,OUT.03
        002454  104202  105704                                                          MOV     #12!ERHARD+3000.,R2
        002456  104020  001035                                                          MOV     R2,OUT.02
        002460  104200  002460  001034                                                  MOV     #.,OUT.01
        002463  104200  060013  001033                                                  MOV     #ERRMES,OUT.RQ
     65 002466  021732                          CALL    TESTEW                   ; BRANCH TO DONE
     66 002467  002607                          BR      READ1X
     67 002470                          2$:
     68 002470  102201  100000                  BIT #RWRDY,R1                    ; SEE IF READ WRITE READY IS STILL HIGH
     69 002472  052515                          BNE READ1C                       ; IF SO, BRANCH
     70 002473                                  HARDER  24                       ;READ/WRITE READY DROPPED BEFORE READ
        002473  104200  001306  001037                                                  MOV     #MS24,OUT.04
        002476  104300  001475  001036                                                  MOV     LUNIT,OUT.03
        002501  104202  105720                                                          MOV     #24!ERHARD+3000.,R2
        002503  104020  001035                                                          MOV     R2,OUT.02
        002505  104200  002505  001034                                                  MOV     #.,OUT.01
        002510  104200  060013  001033                                                  MOV     #ERRMES,OUT.RQ
     71 002513  021740                          CALL    TESTEV                   ; BRANCH
     72 002514  002607                          BR      READ1X
     73 002515  104302  002636          READ1C: MOV     SDI,R2
     74 002517  060012                          XFC WAITSI                       ;WAIT FOR SECTOR OR INDEX PULSE B4 READ
     75 002520  060002                          XFC XREAD                        ;READ THE SECTOR
     76 002521  115001                          TST R1                           ;CHECK FOR ERROR
     77 002522  012607                          BEQ READ1X                       ;END ROUTINE IF READ OK
     78 002523  117403                          DEC R3                           ;COUNT MAX SECTORS TO READ
     79 002524  012537                          BEQ READ1E                       ;REPORT ERROR IF ALL READ
     80 002525  104207  003646                  MOV #RBUF0+RW.LOW,R0
     81 002527  104171                          MOV (R0),R1                      ;INCREMENT BLOCK NUMBER
     82 002530  115401                          INC R1
     83 002531  100271                          MOV R1,(R0)+
     84 002532  042536                          BCC READ1B
     85 002533  104171                          MOV (R0),R1
     86 002534  115401                          INC R1
     87 002535  100171                          MOV R1,(R0)
     88 002536  002412                  READ1B: BR READ1A                        ;GO READ NEXT SECTOR
     89 002537                          READ1E: HARDER  28,<LETTER,RBUF0+RW.LOW,RBUF0+RW.HI,INS+1,INS+2,INS+3,RBUF0+RW.CMD>
        002537  104200  001716  001037                                                  MOV     #MS28,OUT.04
        002542  104300  003102  001040                                                  MOV     LETTER,OUT.05
```

```
UDAT3 DISK FUNCTIONAL DMACR X04.01 13-APR-82 18:08:21 PAGE 34-2
READ1 - READ SECTORS ON A TRACK FOR TEST

        002545  104300  003646  001041                                  MOV     RBUF0+RW.LOW,OUT.06
        002550  104300  003647  001042                                  MOV     RBUF0+RW.HI,OUT.07
        002553  104300  002741  001043                                  MOV     INS+1,OUT.08
        002556  104300  002742  001044                                  MOV     INS+2,OUT.09
        002561  104300  002743  001045                                  MOV     INS+3,OUT.10
        002564  104300  003650  001046                                  MOV     RBUF0+RW.CMD,OUT.11
        002567  104300  001475  001036                                  MOV     LUNIT,OUT.03
        002572  104202  105724                                          MOV     #28!ERHARD+3000.,R2
        002574  104020  001035                                          MOV     R2,OUT.02
        002576  104200  002576  001034                                  MOV     #.,OUT.01
        002601  104200  060013  001033                                  MOV     #ERRMES,OUT.RQ
    90  002604  104307  001033                    MOV     OUT.RQ,R0
    91  002606  021000                          CALL    HOSTRQ
    92  002607                          READ1X: POP <R5,R4,R3,R1,R0>
        002607  104265                                                          MOV (SP)+,R5
        002610  104264                                                          MOV (SP)+,R4
        002611  104263                                                          MOV (SP)+,R3
        002612  104261                                                          MOV (SP)+,R1
        002613  104267                                                          MOV (SP)+,R0
    93  002614  000000                          RETURN
```

```
                                .SBTTL  UNITS, SDI COMMANDS AND PROGRAM VARIABLES
                        ;PROGRAM VARIABLES

                        ;UNIT NUMBER STORAGE FOR DISK DRIVES TO TEST

002615  000020          UNITS:  .REPT   16.                     ; 16 SUBUNITS (8 MAX) 4 ON EACH UNIT
                                .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
                                .ENDR
002615  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002616  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002617  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002620  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002621  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002622  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002623  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002624  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002625  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002626  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002627  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002630  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002631  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002632  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002633  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH
002634  177777                  .WORD   177777                  ; MARK ALL UNITS AS NON-EXISTANT TO START WITH

002635  000000          UNITNB: .WORD 0                         ;NUMBER OF UNIT CURRENTLY UNDER TEST
                                                                ;POINTER TO TABLE ABOVE

002636  000000          SDI:    .WORD 0                         ;SDI INTERCONNECT CODE FOR XFC CALLS

002637  000000          COMND:  .WORD   0                       ; POINTER TO COMMAND MESSAGE
                        ;       MESSAGE TABLES

002640                  CR.ONL: MSG     ONL,2,ST,7,COMPLT       ; DRIVE ONLINE
002640  002722                  .WORD ONL                       ;ADDRESS OF COMMAND
002641  000002                  .WORD 2                 ;SIZE OF COMMAND IN BYTES
002642  002751                  .WORD ST                        ;ADDRESS OF REPLY
002643  000007                  .WORD 7                 ;SIZE OF REPLY IN WORDS
002644  000176                  .WORD COMPLT                    ; SUCCESSFUL COMPLETION CODE
        002644  SNDONE  =       .-1                     ;PREVIOUS COMMANDS CAN BE SENNT TO AN OFFLINE DRIVE
002645                  CR.CLR: MSG     DRC,2,ST,7,COMPLT       ; DRIVE CLEAR
002645  002730                  .WORD DRC                       ;ADDRESS OF COMMAND
002646  000002                  .WORD 2                 ;SIZE OF COMMAND IN BYTES
002647  002751                  .WORD ST                        ;ADDRESS OF REPLY
002650  000007                  .WORD 7                 ;SIZE OF REPLY IN WORDS
002651  000176                  .WORD COMPLT                    ; SUCCESSFUL COMPLETION CODE
002652                  CR.DIS: MSG     DIS,2,ST,7,COMPLT       ; DISCONNECT
002652  002726                  .WORD DIS                       ;ADDRESS OF COMMAND
002653  000002                  .WORD 2                 ;SIZE OF COMMAND IN BYTES
002654  002751                  .WORD ST                        ;ADDRESS OF REPLY
002655  000007                  .WORD 7                 ;SIZE OF REPLY IN WORDS
002656  000176                  .WORD COMPLT                    ; SUCCESSFUL COMPLETION CODE
002657                  CR.GCR: MSG     GCR,1,CR,12.,170        ; GET CHARACTERISTICS
002657  002732                  .WORD GCR                       ;ADDRESS OF COMMAND
002660  000001                  .WORD 1                 ;SIZE OF COMMAND IN BYTES
002661  002760                  .WORD CR                        ;ADDRESS OF REPLY
002662  000014                  .WORD 12.                       ;SIZE OF REPLY IN WORDS
```

```
                002663  000170                  .WORD 170                       ; SUCCESSFUL COMPLETION CODE
     23         002664                  CR.SCR: MSG     SCR,2,SUB,21.,167       ; GET SUBUNIT CHARACTERISTICS
                002664  002733                  .WORD SCR                       ;ADDRESS OF COMMAND
                002665  000002                  .WORD 2                 ;SIZE OF COMMAND IN BYTES
                002666  002773                  .WORD SUB                       ;ADDRESS OF REPLY
                002667  000025                  .WORD 21.                       ;SIZE OF REPLY IN WORDS
                002670  000167                  .WORD 167                       ; SUCCESSFUL COMPLETION CODE
     24         002671                  CR.GST: MSG     GST,1,ST,7,366          ; GET STATUS
                002671  002735                  .WORD GST                       ;ADDRESS OF COMMAND
                002672  000001                  .WORD 1                 ;SIZE OF COMMAND IN BYTES
                002673  002751                  .WORD ST                        ;ADDRESS OF REPLY
                002674  000007                  .WORD 7                 ;SIZE OF REPLY IN WORDS
                002675  000366                  .WORD 366                       ; SUCCESSFUL COMPLETION CODE
     25         002676                  CR.MOD: MSG     MOD,3,ST,7,COMPLT       ; MODE
                002676  002744                  .WORD MOD                       ;ADDRESS OF COMMAND
                002677  000003                  .WORD 3                 ;SIZE OF COMMAND IN BYTES
                002700  002751                  .WORD ST                        ;ADDRESS OF REPLY
                002701  000007                  .WORD 7                 ;SIZE OF REPLY IN WORDS
                002702  000176                  .WORD COMPLT                    ; SUCCESSFUL COMPLETION CODE
     26         002703                  CR.SEK: MSG     INS,6,ST,7,COMPLT       ; INITIATE SEEK
                002703  002740                  .WORD INS                       ;ADDRESS OF COMMAND
                002704  000006                  .WORD 6                 ;SIZE OF COMMAND IN BYTES
                002705  002751                  .WORD ST                        ;ADDRESS OF REPLY
                002706  000007                  .WORD 7                 ;SIZE OF REPLY IN WORDS
                002707  000176                  .WORD COMPLT                    ; SUCCESSFUL COMPLETION CODE
     27         002707                  LONG = .-1                              ; LONG TIMEOUT COMMANDS FOLLOW
     28         002710                  CR.INR: MSG     INR,1,ST,7,COMPLT       ; INITIATE RECALIBRATE
                002710  002737                  .WORD INR                       ;ADDRESS OF COMMAND
                002711  000001                  .WORD 1                 ;SIZE OF COMMAND IN BYTES
                002712  002751                  .WORD ST                        ;ADDRESS OF REPLY
                002713  000007                  .WORD 7                 ;SIZE OF REPLY IN WORDS
                002714  000176                  .WORD COMPLT                    ; SUCCESSFUL COMPLETION CODE
     29         002715                  CR.RUN: MSG     RUN,1,ST,7,COMPLT       ; RUN
                002715  002736                  .WORD RUN                       ;ADDRESS OF COMMAND
                002716  000001                  .WORD 1                 ;SIZE OF COMMAND IN BYTES
                002717  002751                  .WORD ST                        ;ADDRESS OF REPLY
                002720  000007                  .WORD 7                 ;SIZE OF REPLY IN WORDS
                002721  000176                  .WORD COMPLT                    ; SUCCESSFUL COMPLETION CODE
     30
     31                                 ;       LEVEL 2 COMMAND MESSAGE DATA STRUCTURES
     32
     33 002722     000     213          ONL:    .BYTE   0,213                   ; BRING DRIVE ONLINE
     34 002723  000077                          .WORD   77
     35 002724     000     003          DIA:    .BYTE   0,3                     ; DIAGNOSE
     36 002725  000000                          .WORD   0
     37 002726     000     204          DIS:    .BYTE   0,204                   ; DISCONNECT
     38 002727  000000                          .WORD   0
     39 002730     000     005          DRC:    .BYTE   0,DRVCLR                ; DRIVE CLEAR
     40 002731  000000                  ERRORS: .WORD   0                       ; ERROR FLAGS
     41 002732     000     207          GCR:    .BYTE   0,GETCHR                ; GET CHARACTERISTICS
     42 002733     000     210          SCR:    .BYTE   0,GETSUB                ; GET SUBUNIT CHARACTERISTICS
     43 002734  000000                  SUBUNT: .WORD   0                       ; SUBUNIT SELECTION IN LOW ORDER BYTE
     44 002735     000     011          GST:    .BYTE   0,GETSTA                ; GET STATUS
     45 002736     000     014          RUN:    .BYTE   0,14                    ; SPIN UP DRIVE
     46 002737     000     216          INR:    .BYTE   0,IRECLB                ; INITIATE RECALIBRATE
     47 002740     000     012          INS:    .BYTE   0,INSEEK                ; INITIATE SEEK
     48 002741  000000                          .WORD   0                       ; INS CYLINDER/HEAD ARGUMENTS
```

```
     49 002742  000000                          .WORD   0
     50 002743  000000                          .WORD   0
     51 002744     000      201        MOD:     .BYTE   0,201                  ; CHANGE MODE
     52 002745  000000                 MODE:    .WORD   0
     53 002746     362      377        MODE1:   .BYTE   362,377
     54 002747     006      377        MODE2:   .BYTE   6,377
     55
     56 002750     000      014        RUNCMD:  .BYTE   0,14
     57          000014                RUNLBC   =       14                     ; RUN COMMAND
     58
     59                                ;        RESPONSE  MESSAGE DATA BUFFERS
     60
     61 002751                         ST:      .BLKW   7                      ; STATUS MESSAGE BUFFER
     62 002760                         CR:      .BLKW   31.                    ; CHARACTERISTICS MESSAGE BUFF
     63                                ;        ASSUME  SUB,CR+11.
     64
     65 003017  000000                 DMSDI:   .WORD   0                      ; DUMMY SDI CONTROL BLK
     66 003020  000000                 NSCSL:   .WORD   0                      ; # OF SECTORS IN HEADER SEARCH
     67 003021  002766                          .WORD   SUB-5                  ; POINTER TO SUBUNIT CHAR.
     68 003022                                  .BLKW   5                      ; WORD DMSDI+7 IS CLOBBERED BY UDA
     69                                                                        ; SO SET ASIDE SPACE
     70 003027                                  .BLKW   2                      ; RESERVED FOR UDA PRIMARY REVECTORING
     71 003031                                  .BLKW   8.                     ; SKIP SPACE TO POINTER
     72 003041  003013                          .WORD   DMSDI-4                ; PRI REV INFO WRITTEN IN DMSDI+8 AND 9
     73
     74                                ;        DISK LOCATION POINTERS
     75
     76 003042                         ROFDBN:  .BLKW   2                      ; FIRST READ ONLY DBN
     77 003044                         ROFDC:   .BLKW   3                      ; FIRST READ ONLY CYL AND GROUP
     78 003047                         LCDBN:   .BLKW   2                      ; FIRST FACTORY FORMATTED DBN
     79 003051                         LDIACYL: .BLKW   3                      ; FACTORY FORMATTED CYLINDER
     80 003054                         FDIACYL: .BLKW   2                      ; FIRST DIAGNOSTIC CYLINDER
     81 003056                         FXBNCYL: .BLKW   2                      ; FIRST XBN CYLINDER
     82 003060                         SECTRK:  .BLKW   1                      ; SECTORS PER TRACK
     83
     84 003061                         TSTBLK:  .BLKW   2                      ; TEST BLOCK NUMBER
     85 003063                         TSTCYL:  .BLKW   3                      ; TEST CYLINDER NUMBER
     86 003066                         BLOCKT:  .BLKW   1                      ; BLOCKS ON CURRENT TRACK
     87 003067                         BLOCKG:  .BLKW   1                      ; BLOCKS ON CURRENT GROUP
     88 003070                         BLOCKC:  .BLKW   1                      ; BLOCKS ON CURRENT CYL
     89 003071                         CURBLK:  .BLKW   2                      ; CURRENT BLOCK NUMBER
     90 003073                         SECGRP:  .BLKW   1                      ; NUMBER OF SECTORS PER GROUP
     91 003074                         SECCYL:  .BLKW   1                      ; NUMBER OF SECTORS PER CYL
     92 003075                         CURGRP:  .BLKW   1                      ; CURRENT GROUP NUMBER
     93 003076  000000                 CURTRK:  .WORD   0                      ; CURRENT TRACK
     94 003077  000000                 CURPAT:  .WORD   0                      ; CURRENT PATTERN
     95 003100  000000                 SECCNT:  .WORD   0                      ; SECTOR COUNT
     96
     97 003101  000000                 AREA:    .WORD   0                      ; TO STORE AREA LETTER L OR X (BN)
     98 003102  000000                 LETTER:  .WORD   0                      ; TO STORE CURRENT AREA L, D OR X (BN)
     99
    100 003103  000000                 ERRPOS:  .WORD   0
    101 003104  000000                 WFLAG:   .WORD   0                      ; FOR WRITE PROTECTION TEST
    102
    103 003105  000000                 FLAG:    .WORD   0                      ; TEST ALL CYLINDER FLAG
    104 003106  000000                 GRPCNT:  .WORD   0                      ; COUNT OF CYL'S HANDLED
    105 003107  000000                 OLDGRP:  .WORD   0                      ; HOLDS OLD GROUP VALUE FROM EVL COMPUTE
```

```
    106
    107 003110  000000  000000          VAR1:   .WORD   0,0                     ; FOR CONVERT XFC
    108 003112  000000  000000          VAR2:   .WORD   0,0
    109 003114  000000                  VAR3:   .WORD   0
    110 003115  000000                  VAR4:   .WORD   0
    111 003116  000000  000000          CYLLO:  .WORD   0,0
    112 003120  000000                  GROUP:  .WORD   0
    113 003121  000000                  TRACK:  .WORD   0
    114 003122  000000                  SSCTOR: .WORD   0
    115 003123  000000                  INDEXS: .WORD   0
    116
    117                                 ;       WRITE DATA BUFFERS
    118
    119 003124                          CHAIN:                                  ; READ AND WRITE CHAIN AREA
    120 003124  000000                          .WORD   0                       ; STATUS AREA
    121 003125  000000                          .WORD   0                       ; BUFFER POINTER
    122 003126  000000                          .WORD   0                       ; LO ORDER HEADER
    123 003127  000000                          .WORD   0                       ; HI ORDER HEADER
    124 003130  000000                          .WORD   0                       ; REAL TIME COMMAND AND TRACK NUMBER
    125 003131  003017                          .WORD   DMSDI                   ; POINTER TO DUMMY SDI CONTROL BLOCK
    126
```

```
                                                .SBTTL  DATA PATTERNS
003132  000004                          PATPTR: .WORD   4                       ; FOUR PATTERNS WILL BE TESTED
003133  003137                                  .WORD   PAT4
003134  003160                                  .WORD   PAT5
003135  003201                                  .WORD   PAT6
003136  003222                                  .WORD   PAT8
                                        ;
                                        ;       THE DATA PATTERNS (LENGTH OF PATTERN IN WORDS, FOLLOWED BY PATTERN)
                                        ;
003137  000016                          PAT4:   .WORD   16                      ; SHIFTING ONES
003140  000001                                  .WORD   000001
003141  000003                                  .WORD   000003
003142  000007                                  .WORD   000007
003143  000017                                  .WORD   000017
003144  000037                                  .WORD   000037
003145  000077                                  .WORD   000077
003146  000177                                  .WORD   000177
003147  000377                                  .WORD   000377
003150  000777                                  .WORD   000777
003151  001777                                  .WORD   001777
003152  003777                                  .WORD   003777
003153  007777                                  .WORD   007777
003154  017777                                  .WORD   017777
003155  037777                                  .WORD   037777
003156  077777                                  .WORD   077777
003157  177777                                  .WORD   177777
003160  000016                          PAT5:   .WORD   16                      ; SHIFTING ZEROS
003161  177776                                  .WORD   177776
003162  177774                                  .WORD   177774
003163  177770                                  .WORD   177770
003164  177760                                  .WORD   177760
003165  177740                                  .WORD   177740
003166  177700                                  .WORD   177700
003167  177600                                  .WORD   177600
003170  177400                                  .WORD   177400
003171  177000                                  .WORD   177000
003172  176000                                  .WORD   176000
003173  174000                                  .WORD   174000
003174  170000                                  .WORD   170000
003175  160000                                  .WORD   160000
003176  140000                                  .WORD   140000
003177  100000                                  .WORD   100000
003200  000000                                  .WORD   000000
003201  000016                          PAT6:   .WORD   16                      ; ALTERNATING ZERO WORD AND ONE WORD IN
003202  000000                                  .WORD   000000                  ; 3-2-1-1-1
003203  000000                                  .WORD   000000
003204  000000                                  .WORD   000000
003205  177777                                  .WORD   177777
003206  177777                                  .WORD   177777
003207  177777                                  .WORD   177777
003210  000000                                  .WORD   000000
003211  000000                                  .WORD   000000
003212  177777                                  .WORD   177777
003213  177777                                  .WORD   177777
003214  000000                                  .WORD   000000
003215  177777                                  .WORD   177777
003216  000000                                  .WORD   000000
```

```
     58 003217  177777                          .WORD   177777
     59 003220  000000                          .WORD   000000
     60 003221  177777                          .WORD   177777
     61 003222  000016              PAT8:       .WORD   16                      ; B'0101010101010101' AND
     62 003223  052525                          .WORD   052525                  ; B'1010101010101010'
     63 003224  052525                          .WORD   052525                  ; IN 3-2-1-1-1
     64 003225  052525                          .WORD   052525
     65 003226  125252                          .WORD   125252
     66 003227  125252                          .WORD   125252
     67 003230  125252                          .WORD   125252
     68 003231  052525                          .WORD   052525
     69 003232  052525                          .WORD   052525
     70 003233  125252                          .WORD   125252
     71 003234  125252                          .WORD   125252
     72 003235  052525                          .WORD   052525
     73 003236  125252                          .WORD   125252
     74 003237  052525                          .WORD   052525
     75 003240  125252                          .WORD   125252
     76 003241  052525                          .WORD   052525
     77 003242  125252                          .WORD   125252
     78
     79
     80 003243                      OBUFF:      .BLKW   257.                    ; WRITE DATA BUFFER
     81
     82                             ;           READ DATA BUFFERS
     83
     84 003644                      RBUF0:                                      ; FIRST BUFFER
     85 003644  100000                          .WORD   RSTOP                   ; STATUS AND LINK POINTER
     86 003645  003652                          .WORD   RBUFD                   ; POINTER TO DATA
     87 003646                                  .BLKW   2
     88 003650  013400                          .WORD   RREAL                   ; LEVEL 1 SDI COMMAND
     89 003651  003017                          .WORD   DMSDI                   ; POINTER TO DUMMY SDI CONTROL BLOCK
     90
```

```
                                        .SBTTL  START, GETU<POLL ALL PORTS>, RBUFD, & FCHAIN

                                .IF     GE,<.+<255.*3>>-7774
                                        .ERROR
                                .ENDC

003652                          RBUFD:; .BLKW   274.                    ; READ DATA BUFFER

                                ;       FORMAT CHAIN

003652                          FCHAIN:
                                ;       .REPT   255.                    ; UP TO 240+ DBNS/TRK MAXIMUM
                                ;       .WORD   0                       ; BUFFER POINTER
                                ;       .WORD   0                       ; LO ORDER DBN
                                ;       .WORD   0                       ; HI ORDER DBN
                                ;       .ENDR
                                ;       .WORD   0                       ; EXTRA WORD FOR END-OF-CHAIN FLAG
003652                          START:
003652                          GETU:
                                        .SBTTL  GETU   - POLL ALL PORTS, THEN GET UNITS TO TEST
                                ;
                                ;       GET THE UNITS TO TEST
                                ;
                                ;
                                ;       POLL ALL PORTS AND FILL IN A UDA PORT INFORMATION TABLE (UNITS)
                                ;
003652  114000  001730                  CLR     OVRLNM                  ; CLEAR OVERLAY NUMBER TO INITATE
                                ; *** SET UP OVERLAY TABLE
003654  104302  007774                  MOV     OVSTRT,R2               ; GET STARTING ADDRESS OF OVERLAY (LO)
003656  104303  007775                  MOV     OVSTRT+1,R3             ; GET STARTING ADDRESS OF OVERLAY (HI)
003660  104020  001547                  MOV     R2,OTABLE               ; MOVE STARTING ADDRESS OF OVERLAY
003662  104207  000002                  MOV     #NUMOVL-1,R0            ; R0 = NUMBER OF OVERLAYS
003664  104205  001547                  MOV     #OTABLE,R5              ; R5 -> OVERLAY
003666  104651  000001          1$:     MOV     1(R5),R1                ; R1 HAS OVER LAY LENGTH
003670  110601                          ROR     R1                      ; SHIFT TO MAKE BYTES
003671  102201  100001                  BIT     #100001,R1              ; CLEAR UNUSED BITS
003673  105012                          ADD     R1,R2                   ; FIND ADDRESS OF NEXT OVERLAY
003674  043676                          BCC     2$                      ; IF NO CARRY BRANCH
003675  115403                          INC     R3                      ; PROPOGATE CARRY
003676  100652  000002          2$:     MOV     R2,2(R5)                ; STORE STARTING ADDRESS OF NEXT OVERLAY
003700  104651  000003                  MOV     3(R5),R1                ; GET OVERLAY LENGTH (<<2)
003702  101031                          BIS     R3,R1                   ; SAVE HI ORDER OVERLAY ADDRESS
003703  100651  000003                  MOV     R1,3(R5)                ; SAVE
003705  105205  000002                  ADD     #2,R5                   ; POINT TO NEXT OVERLAY AREA
003707  117407                          DEC     R0                      ; DECREMENT COUNT
003710  053666                          BNE     1$                      ; IF ALL OVERLAYS NOT SET UP, BRANCH
                                ; *** NOW DO REGULAR GETU
003711  104205  000001                  MOV     #1,R5                   ; MOVE INITIAL MASK TO R5
003713  104204  002615                  MOV     #UNITS,R4               ; R4 POINTS TO UNIT TABLE
003715                          5$:     PUSH    R4                      ; SAVE R4
003715  100464                                                                          MOV R4,-(SP)
003716  104052                          MOV     R5,R2                   ; MOVE MASK TO R2
003717  020754                          CALL    RDSTAT                  ; GET DRIVE'S STATUS
003720  115002                          TST     R2                      ; SEE IF ERROR
003721  013727                          BEQ     10$                     ; IF NOT, BRANCH
003722  104203  002626                  MOV     #SER10,R3               ; NO DRIVE ATTACHED
003724  100643  000001                  MOV     R3,1(R4)                ; SAVE ERROR MESSAGE
```

```
     57 003726  004120                      BR      85$             ; REPORT
     58 003727  114003              10$:    CLR     R3              ; SET UP TIMEOUT COUNT
     59 003730  104052              15$:    MOV     R5,R2           ; MOVE MASK TO R2
     60 003731  020754                      CALL    RDSTAT          ; GET STATUS
     61 003732  102201  000001              BIT     #RCVRDY,R1      ; SEE IF RECEIVER READY ASSERTED
     62 003734  053744                      BNE     20$             ; IF SO, BRANCH
     63 003735  117403                      DEC     R3              ; DECREMENT COUNT
     64 003736  053730                      BNE     15$             ; IF INCOMPLETE, BRANCH
     65 003737  104203  002641              MOV     #SER11,R3       ; RECEIVER READY NEVER ASSERTED
     66 003741  100643  000001              MOV     R3,1(R4)        ; SAVE ERROR MESSAGE
     67 003743  004120                      BR      85$             ; REPORT
     68 003744  102201  000100      20$:    BIT     #AVAIL,R1       ; SEE IF DRIVE IS AVAILABLE
     69 003746  053754                      BNE     25$             ; IF SO, BRANCH
     70 003747  104203  003134              MOV     #SER40,R3       ; GET SECONDARY ERROR
     71 003751  100643  000001              MOV     R3,1(R4)        ; SAVE
     72 003753  004120                      BR      85$             ; EXIT
     73 003754  104202  001750      25$:    MOV     #MAXSND,R2      ; SET UP MAXIMUM TRIES AT SENDING
     74 003756  104203  002671              MOV     #CR.GST,R3      ; R3 POINTS TO GET STATUS COMMAND
     75 003760  104137              30$:    MOV     (R3),R0         ; SET ADR OF SDI COMMAND BUFFER
     76 003761  104631  000001              MOV     1(R3),R1        ; SET BUFFER LENGTH
     77 003763                              PUSH    R2              ; SAVE R2
        003763  100462                                                                  MOV R2,-(SP)
     78 003764  104052                      MOV     R5,R2           ; SETUP FOR SEND
     79 003765  060004                      XFC     SEND            ; SEND COMMAND
     80 003766                              POP     R2              ; RESTORE COUNT
        003766  104262                                                                  MOV (SP)+,R2
     81 003767  115001                      TST     R1              ; DID UNIT ACCEPT COMMAND
     82 003770  014000                      BEQ     35$             ; IF SO, BRANCH
     83 003771  117402                      DEC     R2              ; DECREMENT COUNT
     84 003772  053760                      BNE     30$             ; IF UNEXPIRED, BRANCH
     85 003773  104203  002657              MOV     #SER12,R3       ; GET ERROR NUMBER
     86 003775  100643  000001              MOV     R3,1(R4)        ; SAVE
     87 003777  004120                      BR      85$
     88 004000                      35$:    PUSH    R4              ; SAVE R4
        004000  100464                                                                  MOV R4,-(SP)
     89 004001  104204  000003              MOV     #3,R4           ; SET UP SHORT TIMEOUT
     90 004003  104207  002751      40$:    MOV     #ST,R0          ; SET DATA BUFFER ADDRESS
     91 004005  104631  000003              MOV     3(R3),R1        ; SET BUFFER LENGTH
     92 004007  104052                      MOV     R5,R2           ; SETUP FOR RECEIVE
     93 004010  060005                      XFC     RCV             ; RECEIVE SDI COMMAND
     94 004011  115001                      TST     R1              ; DID ERROR OCCUR
     95 004012  014052                      BEQ     70$             ; IF NOT, BRANCH
     96 004013  106201  000001              CMP     #1,R1           ; SEE IF TIMEOUT
     97 004015  054024                      BNE     45$             ; IF NOT, BRANCH
     98 004016  117404                      DEC     R4              ; DECREMENT TIMEOUT VALUE
     99 004017  054003                      BNE     40$             ; IF NOT TIMEOUT, BRANCH
    100 004020                              POP     R4              ; RESTORE R4
        004020  104264                                                                  MOV (SP)+,R4
    101 004021  104203  002671              MOV     #SER13,R3       ; GET ERROR NUMBER
    102 004023  004047                      BR      65$             ; BRANCH TO EXIT
    103 004024                      45$:    POP     R4              ; RESTORE R4
        004024  104264                                                                  MOV (SP)+,R4
    104 004025  110601                      ROR     R1              ; ROTATE INTO POSITION TO TEST
    105 004026  110601                      ROR     R1              ; SEE IF FIRST WORD NOT START FRAME
    106 004027  044033                      BCC     50$             ; IF NOT, BRANCH
    107 004030  104203  002704              MOV     #SER14,R3       ; GET ERROR NUMBER
    108 004032  004047                      BR      65$             ; BRANCH TO END OF LOOP
```

```
    109 004033  110601                50$:    ROR     R1              ; SEE IF FRAMING ERROR
    110 004034  044040                        BCC     55$             ; IF NOT, BRANCH
    111 004035  104203  002732                MOV     #SER15,R3       ; GET ERROR NUMBER
    112 004037  004047                        BR      65$             ; BRANCH TO END OF LOOP
    113 004040  110601                55$:    ROR     R1              ; SEE IF CHECKSUM ERROR
    114 004041  044045                        BCC     60$             ; IF NOT, BRANCH
    115 004042  104203  002754                MOV     #SER16,R3       ; GET ERROR NUMBER
    116 004044  004047                        BR      65$             ; BRANCH TO END OF LOOP
    117 004045  104203  002777        60$:    MOV     #SER17,R3       ; GET ERROR NUMBER
    118 004047  100643  000001        65$:    MOV     R3,1(R4)        ; SAVE
    119 004051  004120                        BR      85$             ; BRANCH TO END OF LOOP
```

```
                                        ;
                                        ;       NOW FILL IN THE TABLE WITH ALL THE SUBUNIT NUMBERS
                                        ;
004052                          70$:    POP     R4                      ; RESTORE R4
004052  104264                                                                 MOV (SP)+,R4
004053  104207  177777                  MOV     #-1,R0                  ; GET 'NO UNITS' FLAG
004055  100647  000001                  MOV     R0,1(R4)                ; CLEAR ANY ERRORS THAT ARE FLAGGED
004057  100647  000002                  MOV     R0,2(R4)                ; CLEAR ANY ERRORS THAT ARE FLAGGED
004061  104307  002751                  MOV     ST,R0                   ; R0 HAS UNIT NUMBER
004063  104072                          MOV     R0,R2                   ; COPY R0 TO R2
004064  103207  170000                  BIC     #^CHBHINB,R0            ; R0 HAS UNIT NUMBER
004066                                  PUSH    <R1,R2>                 ; SAVE R1 AND R2
004066  100461                                                                  MOV R1,-(SP)
004067  100462                                                                  MOV R2,-(SP)
004070  104052                          MOV     R5,R2                   ; MOVE UDA PORT MASK TO R2
004071  020754                          CALL    RDSTAT                  ; GET STATUS
004072  102201  000002                  BIT     #ATTN,R1                ; SEE IF SPINABLE
004074  054077                          BNE     75$                     ; IF SO, BRANCH
004075  101207  010000                  BIS     #10000,R0               ; SET 'NOT SPINABLE' FLAG
004077                          75$:    POP     <R2,R1>                 ; RESTORE
004077  104262                                                                  MOV (SP)+,R2
004100  104261                                                                  MOV (SP)+,R1
004101  101207  040000                  BIS     #40000,R0               ; FLAG UNIT AS NOT TESTED
004103  110702                          SWAB    R2                      ; SWAP R2'S BYTES
004104  110602                          ROR     R2                      ; MOVE SUBUNIT MASK TO LO NIBBLE
004105  110602                          ROR     R2                      ; MOVE SUBUNIT MASK TO LO NIBBLE
004106  110602                          ROR     R2                      ; MOVE SUBUNIT MASK TO LO NIBBLE
004107  110602                          ROR     R2                      ; MOVE SUBUNIT MASK TO LO NIBBLE
004110  103202  177760                  BIC     #L8LONB,R2              ; CLEAR ALL BUT SUBUNIT BITS
004112  100147                          MOV     R0,(R4)                 ; SET UNIT NUMBER IN UNITS
004113  110602                  80$:    ROR     R2                      ; MOVE SUBUNIT BIT TO CARRY
004114  044120                          BCC     85$                     ; IF NO MORE SUBUNITS, BRANCH
004115  100247                          MOV     R0,(R4)+                ; MOVE SUBUNIT NUMBER TO TABLE
004116  115407                          INC     R0                      ; INCREMENT SUBUNIT NUMBER
004117  004113                          BR      80$                     ; BRANCH
                                        ;
                                        ;
                                        ;
004120                          85$:    POP     R4                      ; R4 POINTS TO START OF UNIT JUST HANDLED
004120  104264                                                                 MOV (SP)+,R4
004121  105204  000004                  ADD     #4,R4                   ; R4 WILL POINT TO NEXT UNIT (CLEAR CARRY FOR ROL)
004123  110205                          ROL     R5                      ; R5 HAS NEXT UNIT PORT MASK
004124  106205  000020                  CMP     #20,R5                  ; SEE IF ALL PORTS TESTED
004126  053715                          BNE     5$                      ; IF NOT, BRANCH
```

```
                                        ;
                                        ;       NOW GET THE PLUG NUMBERS TO TEST AND FIND THEM IN THE TABLE
                                        ;
004127  104207  060012                  MOV     #UTOTST,R0      ; GET WHAT SUBUNIT NUMBERS TO TEST REQUEST
004131  021000                          CALL    HOSTRQ          ; GET THE PLUG NUMBERS
004132  104207  001034                  MOV     #OUT.01,R0      ; R0 POINTS TO UNIT NUMBERS TO TEST
004134  104201  002615          90$:    MOV     #UNITS,R1       ; R1 POINTS TO UDA PORT INFORMATION
004136  104112                  95$:    MOV     (R1),R2         ; R2 HAS UNIT
004137  074162                          BMI     115$            ; IF NONE, BRANCH
004140                                  PUSH    R1              ; SAVE R1
004140  100461                                                                  MOV R1,-(SP)
004141  103202  160000          100$:   BIC     #160000,R2      ; CLEAR 'NOT TESTED', LEAVE 'UNSPINABLE' SET
004143  106172                          CMP     (R0),R2         ; SEE IF IT IS A UNIT TO TEST
004144  054150                          BNE     105$            ; NO MATCH
004145  100112                          MOV     R2,(R1)         ; SAVE UNIT AS ONE TO TEST
004146                                  POP     R1              ; RESTORE STACK
004146  104261                                                                  MOV (SP)+,R1
004147  004267                          BR      160$            ; LOOK FOR THE NEXT ONE
004150  115401          105$:   INC     R1              ; POINT TO NEXT SUBUNIT
004151  104012                          MOV     R1,R2           ; COPY TO R2
004152  107202  002615                  SUB     #UNITS,R2       ; SUBTRACT STARTING ADDRESS
004154  102202  000003                  BIT     #3,R2           ; SEE IF STILL ON SAME UNIT
004156  014161                          BEQ     110$            ; IF NOT, BRANCH
004157  104112                          MOV     (R1),R2         ; SEE IF ANY MORE SUBUNITS
004160  034141                          BPL     100$            ; IF SO, BRANCH
004161                          110$:   POP     R1              ; RESTORE R1
004161  104261                                                                  MOV (SP)+,R1
004162  105201  000004          115$:   ADD     #4,R1           ; LOOK AT NEXT UNIT
004164  106201  002635                  CMP     #UNITS+16.,R1   ; SEE IF ENTIRE TABLE SEARCHED
004166  054136                          BNE     95$             ; IF NOT, BRANCH
```

```
                                        ;
                                        ;       DIDN'T FIND THE REQUESTED UNITS -- DUMP ALL KNOWLEDGE OF THE
                                        ;       UDA PORTS AND DIE
                                        ;
004167  104170  001036                  MOV     (R0),OUT.03     ; SAVE UNIT NUMBER IN REQUEST BUFFER
004171  104204  001040                  MOV     #OUT.05,R4      ; R4 POINTS TO OUTPUT BUFFER
004173  104205  002615                  MOV     #UNITS,R5       ; R5 POINTS TO UNIT TABLE
004175  104157                  120$:   MOV     (R5),R0         ; GET FIRST WORD OF UNIT
004176  034203                          BPL     125$            ; IF VALID UNIT WAS FOUND, BRANCH
004177  104657  000001                  MOV     1(R5),R0        ; GET POINTER TO ERROR MESSAGE
004201  100247                          MOV     R0,(R4)+        ; SAVE IN OUTPUT BUFFER
004202  004236                          BR      155$            ; EXIT
004203                          125$:   PUSH    R5              ; SAVE POINTER TO UNIT TABLE
004203  100465                                                                  MOV R5,-(SP)
004204  102207  010000                  BIT     #10000,R0       ; SEE IF DRIVE UNSPINABLE
004206  014212                          BEQ     130$            ; IF SPINABLE, BRANCH
004207  104207  003156                  MOV     #SER41,R0       ; REPORT DRIVE(S) UNSPINABLE
004211  004214                          BR      135$            ; BRANCH
004212  104207  003031          130$:   MOV     #SER18,R0       ; GET POINTER TO ERROR MESSAGE
004214  100247                  135$:   MOV     R0,(R4)+        ; SAVE
004215  114007                          CLR     R0              ; CLEAR COUNT
004216  104251                  140$:   MOV     (R5)+,R1        ; GET UNIT NUMBER
004217  074224                          BMI     145$            ; IF INVALID, BRANCH
004220  115407                          INC     R0              ; INCREMENT COUNT
004221  106207  000004                  CMP     #4,R0           ; SEE IF MAX
004223  054216                          BNE     140$            ; IF NOT, LOOP
004224  104671  001542          145$:   MOV     SER18E-1(R0),R1 ; GET POINTER TO CORRECT ERROR MESSAGE
004226  100241                          MOV     R1,(R4)+        ; MOVE INTO OUTPUT BUFFER
004227                                  POP     R5              ; RESTORE R5
004227  104265                                                                  MOV (SP)+,R5
004230                                  PUSH    R5              ; SAVE R5
004230  100465                                                                  MOV R5,-(SP)
004231  104251                  150$:   MOV     (R5)+,R1        ; GET SUBUNIT NUMBER
004232  100241                          MOV     R1,(R4)+        ; SAVE
004233  117407                          DEC     R0              ; DECREMENT COUNT
004234  054231                          BNE     150$            ; IF COUNT INCOMPLETE, BRANCH
004235                                  POP     R5              ; RESTORE R5
004235  104265                                                                  MOV (SP)+,R5
004236  105205  000004          155$:   ADD     #4,R5           ; POINT TO NEXT UNIT TABLE
004240  106205  002635                  CMP     #UNITS+16.,R5   ; SEE IF ENTIRE TABLE SEARCHED
004242  054175                          BNE     120$            ; IF NOT, BRANCH
004243                                  DEVFTL  2000            ; REPORT FATAL ERROR (WILL SHOW UP AS A 5000)
004243  104200  002510  001037                                                  MOV     #MS2000,OUT.04
004246  104300  001475  001036                                                  MOV     LUNIT,OUT.03
004251  104202  051610                                                          MOV     #2000!FTLDEV+3000.,R2
004253  104020  001035                                                          MOV     R2,OUT.02
004255  104200  004255  001034                                                  MOV     #.,OUT.01
004260  104200  060013  001033                                                  MOV     #ERRMES,OUT.RQ
004263  104307  001033                  MOV     OUT.RQ,R0       ; SET UP FOR REPORT
004265  021000                          CALL    HOSTRQ          ; REPORT ERROR
004266  001627                          BR      DONECD          ; END TEST
                                        ;
004267  115407                  160$:   INC     R0              ; POINT TO NEXT UNIT TO TEST
004270  104172                          MOV     (R0),R2         ; CHECK NEXT UNIT
004271  034134                          BPL     90$             ; FIND IN UDA PORT INFORMATION
004272  001555                          BR      T3STRT          ; START TEST
004273                          ENDGTU:
```

```
                                        .SBTTL  REST OF FORMAT CHAIN

004273                                  .BLKW   <<255.*3>-<ENDGTU-GETU>>

        005247                          ENDPNT = .

                                ;       OVERLAYS START HERE!!!

        005247                  OVRLPT  =       .
```

```
                                        .SBTTL  MESSAGES
                                ;MESSAGE STORAGE OVERLAY


005247                                  DMOVLY MS,0
005247  064156                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                        .NLIST BEX

000000  042  124  111  MS1:    .ASCII\"TIME-OUT ON SEND'N\
000011  122  061  122          .ASCII\R1R1\
000013  000                    .BYTE 0
000014  042  124  111  MS2:    .ASCII\"TIME-OUT ON RECEIVE'N\
000027  122  061  122          .ASCII\R1R1\
000031  000                    .BYTE 0
000032  042  106  111  MS3:    .ASCII\"FIRST WORD RECEIVED WAS NOT A START FRAME'N\
000060  122  061  122          .ASCII\R1R1\
000062  000                    .BYTE 0
000063  042  106  122  MS4:    .ASCII\"FRAMING ERROR ON LEVEL 0 RESPONSE'N\
000105  122  061  122          .ASCII\R1R1\
000107  000                    .BYTE 0
000110  042  103  110  MS5:    .ASCII\"CHECKSUM ERROR ON LEVEL 0 RESPONSE'N\
000132  122  061  122          .ASCII\R1R1\
000134  000                    .BYTE 0
000135  042  122  105  MS6:    .ASCII\"RESPONSE LONGER THAN EXPECTED'N\
000155  122  061  122          .ASCII\R1R1\
000157  000                    .BYTE 0
000160  042  103  117  MS7:    .ASCII\"CODE FROM RECEIVE WAS UNINTELLIGIBLE FROM SUBSYSTEM = '%16N\
000216  122  061  122          .ASCII\R1R1\
000220  000                    .BYTE 0
000221  042  103  117  MS8:    .ASCII\"COMMAND DID NOT RETURN EXPECTED RESPONSE CODE'N\
000251  122  061               .ASCII\R1\
000252  042  040*  040         .ASCII\"  EXPECTED RESPONSE  '%8N\
000267  042  040   040         .ASCII\"  ACTUAL RESPONSE    '%8N\
000304  122  061               .ASCII\R1\
000305  000                    .BYTE 0
000306  042  104  122  MS9:    .ASCII\"DRIVE NOT ASSERTING RECEIVER READY IN DRIVE STATE'N\
000340  000                    .BYTE 0
000341  042  106  101  MS10:   .ASCII\"FAILED TO RECEIVE VALID DRIVE STATE'NR1\
000365  000                    .BYTE 0
000366  042  103  101  MS11:   .ASCII\"CANNOT RECEIVE DRIVE STATE FROM DRIVE'N\
000412  042  103  110          .ASCII\"CHECK IF DRIVE IS POWERED ON.'NR1\
000433  000                    .BYTE 0
000434  042  104  122  MS12:   .ASCII\"DRIVE STATE RECEIVED HAS BAD PARITY'NR1\
000460  000                    .BYTE 0
000461  042  116  117  MS13:   .ASCII\"NO VALID STATE FROM DRIVE'NR1\
000500  000                    .BYTE 0
000501  042  123  125  MS14:   .ASCII  \"SUBUNIT CHARACTERISTICS SAY THERE ARE ZERO READ ONLY GROUPS'N\
000540  042  111  116          .ASCII  \"IN THE DIAGNOSTIC AREA'N\
000554  000                    .BYTE   0
000555  042  123  125  MS15:   .ASCII  \"SUBUNIT CHARACTERISTICS SAY THERE ARE LESS THAN 1 READ/WRITE'N\
000614  042  107  122          .ASCII  \"GROUPS IN THE DIAGNOSTIC AREA'N\
000634  000                    .BYTE   0
000635  042  116  105  MS16:   .ASCII\"NEITHER R/W READY NOR ATTENTION SET AFTER RECALIBRATE COMMAND'NR1\
000676  000                    .BYTE 0
000677  042  123  125  MS17:   .ASCII\"SUBUNIT CHARACTERISTICS SAY LESS THAN 1 DIAGNOSTIC CYLINDER'N\
000736  000                    .BYTE 0
000737  042  122  105  MS18:   .ASCII\"READ/WRITE READY DROPPED BEFORE FORMAT OPERATION'N\
```

```
 57 000770  000                          .BYTE 0
 58 000771  042  106  117  MS19:        .ASCII\'FORMAT OPERATION REPORTED TIME-OUT FAILURE'N\
 59 001017  042  040  040               .ASCII\'  CYLINDER 'D28'.   GROUP  'D8'.  TRACK 'D8'.'N\
 60 001047  000                          .BYTE 0
 61 001050  042  101  106  MS20:        .ASCII\'AFTER RECAL, ERROR BITS WERE SET'NR1\
 62 001072  000                          .BYTE 0
 63 001073  116  042  114  MS21:        .ASCII\N'LOGGABLE INFORMATION AFTER RECAL'NR1\
 64 001116  000                          .BYTE 0
 65 001117  042  122  105  MS22:        .ASCII\'READ/WRITE READY DROPPED BEFORE WRITE OPERATION'N\
 66 001150  000                          .BYTE 0
 67 001151  042  103  117  MS23:        .ASCII\'COULD NOT WRITE AND READ ANY BLOCK ON THIS TRACK. ON LAST BLOCK:'N\
 68 001212  042  127  122               .ASCII\'WRITE OPERATION REPORTED FAILURE -- ERROR CODE 'D8' OCTAL.'N\
 69 001251  042  104  102               .ASCII\'DBN 'D24'.  CYLINDER 'D28'.  GROUP 'D8'.  TRACK 'D8'.'N\
 70 001305  000                          .BYTE 0
 71 001306  042  122  105  MS24:        .ASCII\'READ/WRITE READY DROPPED BEFORE READ OPERATION'N\
 72 001336  000                          .BYTE 0
 73 001337  042  103  117  MS25:        .ASCII\'COULD NOT WRITE AND READ ANY BLOCK ON THIS TRACK. ON LAST BLOCK:'N\
 74 001400  042  122  105               .ASCII\'READ OPERATION REPORTED FAILURE -- ERROR CODE 'D8' OCTAL.'N\
 75 001436  042  104  102               .ASCII\'DBN 'D24'.  CYLINDER 'D28'.  GROUP 'D8'.  TRACK 'D8'.'N\
 76 001472  000                          .BYTE 0
 77 001473  042  103  117  MS26:        .ASCII\'COULD NOT WRITE AND READ ANY BLOCK ON THIS TRACK. ON LAST BLOCK:'N\
 78 001534  042  104  101               .ASCII\'DATA COMPARE FAILURE ON WORD 'D16'.'N\
 79 001557  042  105  130               .ASCII\'EXPECTED DATA  'H16N\
 80 001572  042  101  103               .ASCII\'ACTUAL DATA    'H16N\
 81 001604  042  103  131               .ASCII\'CYLINDER 'D28'.  GROUP 'D8'.  TRACK 'D8'.'N\
 82 001632  000                          .BYTE 0
 83 001633  042  123  105  MS27:        .ASCII\'SEEK COMPLETE TIME-OUT -- READ/WRITE READY DID NOT SET.'N\
 84 001670  042  123  105               .ASCII\'SEEK WAS TO CYLINDER 'D28'.  GROUP 'D8'.'N\
 85 001715  000                          .BYTE 0
 86 001716  042  116  117  MS28:        .ASCII\'NO BLOCK ON THIS TRACK CAN BE READ. LAST BLOCK TRIED:'N\
 87 001752  101  061  042               .ASCII\A1'BN 'D24'.  CYLINDER 'D28'.  GROUP 'D8'.  TRACK 'D8'.'N\
 88 002006  000                          .BYTE 0
 89 002007  042  101  126  MS29:        .ASCII  \'AVAILABLE WAS NOT ASSERTED AFTER DISCONNECT'N\
 90 002036  042  040  040               .ASCII  \'  STATE RECEIVED  'H16N\
 91 002052  000                          .BYTE   0
 92 002053  042  111  116  MS30:        .ASCII\'INVALID COMMAND 'H16' WAS SUCCESSFUL'N\
 93 002076  000                          .BYTE   0
 94 002077  042  103  117  MS31:        .ASCII\'COMMAND WITH 'R1' LENGTH = 'D8' WAS SUCCESSFUL'N\
 95 002127  000                          .BYTE   0
 96 002130  042  125  116  MS32:        .ASCII\'UNIT DID NOT REPORT TRANSMITTION ERROR'NR1\
 97 002155  000                          .BYTE   0
 98 002156  042  125  116  MS33:        .ASCII\'UNIT ACCEPTED AN INVALID GROUP NUMBER FROM GROUP SELECT LEVEL 1'N\
 99 002217  000                          .BYTE   0
100 002220  042  125  116  MS34:        .ASCII\'UNABLE TO CORRECTLY READ OVERLAY 'D3NR1\
101 002244  000                          .BYTE   0
102 002245  042  123  125  MS35:        .ASCII\'SUCCESSFULLY WROTE IN DBN AREA WHEN DRIVE WAS WRITE PROTECTED'N\
103 002305  000                          .BYTE   0
104 002306  042  124  110  SER39:       .ASCII\'THIS UDA AND ALL DRIVES ATTACHED WILL BE REMOVED FROM TESTING'N\
105 002346  000                          .BYTE   0
106 002347  042  103  117  SER00:       .ASCII\'COMMAND WAS 'R1\
107 002357  000                          .BYTE   0
108 002360  042  117  116  MS.ONL:      .ASCII\'ONLINE'N\
109 002364  000                          .BYTE   0
110 002365  042  104  122  MS.CLR:      .ASCII\'DRIVE CLEAR'N\
111 002374  000                          .BYTE   0
112 002375  042  104  111  MS.DIS:      .ASCII\'DISCONNECT'N\
113 002403  000                          .BYTE   0
```

```
    114 002404      042     107     105  MS.GCR: .ASCII\'GET COMMON CHARACTERISTICS'N\
    115 002422      000                          .BYTE   0
    116 002423      042     107     105  MS.SCR: .ASCII\'GET SUBUNIT CHARACTERISTICS'N\
    117 002442      000                          .BYTE   0
    118 002443      042     107     105  MS.GST: .ASCII\'GET STATUS'N\
    119 002451      000                          .BYTE   0
    120 002452      042     103     110  MS.MOD: .ASCII\'CHANGE MODE'N\
    121 002461      000                          .BYTE   0
    122 002462      042     123     105  MS.SEK: .ASCII\'SEEK'N\
    123 002465      000                          .BYTE   0
    124 002466      042     111     116  MS.INR: .ASCII\'INITIATE RECALIBRATE'N\
    125 002501      000                          .BYTE   0
    126 002502      042     123     120  MS.RUN: .ASCII\'SPIN UP'N\
    127 002507      000                          .BYTE   0
    128 002510      042     125     116  MS2000: .ASCII  \'UNABLE FIND REQUESTED DRIVE FOR TESTING'N\
    129 002535      042     124     110          .ASCII  \'THE FOLLOWING IS VISIBLE ON THE PORTS'N\
    130 002561      042     125     104          .ASCII  \'UDA PORT 0 -- 'R1\
    131 002572      042     125     104          .ASCII  \'UDA PORT 1 -- 'R1\
    132 002603      042     125     104          .ASCII  \'UDA PORT 2 -- 'R1\
    133 002614      042     125     104          .ASCII  \'UDA PORT 3 -- 'R1\-
    134 002625      000                          .BYTE   0
    135 002626      042     116     117  SER10:  .ASCII  \'NO DRIVE ATTACHED'N\
    136 002640      000                          .BYTE   0
    137 002641      042     122     103  SER11:  .ASCII  \'RCVR RDY NEVER ASSERTED'N\
    138 002656      000                          .BYTE   0
    139 002657      042     124     111  SER12:  .ASCII  \'TIMEOUT OF SEND'N\
    140 002670      000                          .BYTE   0
    141 002671      042     124     111  SER13:  .ASCII  \'TIMEOUT OF RECEIVE'N\
    142 002703      000                          .BYTE   0
    143 002704      042     106     111  SER14:  .ASCII  \'FIRST WORD RECEIVED WAS NOT START FRAME'N\
    144 002731      000                          .BYTE   0
    145 002732      042     106     122  SER15:  .ASCII  \'FRAMING ERROR ON LEVEL 0 RECEIVE'N\
    146 002753      000                          .BYTE   0
    147 002754      042     103     110  SER16:  .ASCII  \'CHECKSUM ERROR ON LEVEL 0 RECEIVE'N\
    148 002776      000                          .BYTE   0
    149 002777      042     122     105  SER17:  .ASCII  \'RESPONSE LONGER THAN EXPECTED FOR GET STATUS CMD'N\
    150 003030      000                          .BYTE   0
    151 003031      042     104     122  SER18:  .ASCII  \'DRIVE 'R1\
    152 003036      000                          .BYTE   0
    153 003037      104     066     042  SER18D: .ASCII  \D6'', '\
    154 003042      104     066     042  SER18C: .ASCII  \D6'', '\
    155 003045      104     066     042  SER18B: .ASCII  \D6'', '\
    156 003050      104     066     116  SER18A: .ASCII  \D6N\
    157 003051      000                          .BYTE   0
    158 003052      042     122     105  SER22:  .ASCII\'REAL TIME STATE 'H16N\
    159 003065      042     123     124          .ASCII\'STATUS (R TO L): 'H16S2H16S2H16S2H16S2H16S2H16S2H16N\
    160 003117      000                          .BYTE   0
    161 003120      042     122     105  SER23:  .ASCII\'REAL TIME STATE 'H16N\
    162 003133      000                          .BYTE   0
    163 003134      042     104     122  SER40:  .ASCII  \'DRIVE NOT AVAILABLE TO THIS UDA'N\
    164 003155      000                          .BYTE   0
    165 003156      042     104     122  SER41:  .ASCII  \'DRIVE NOT SPINABLE'N\
    166 003170      000                          .BYTE   0
    167 003171      042     103     117  SER50:  .ASCII\'COMMAND'\
    168 003175      000                          .BYTE   0
    169 003176      042     122     105  SER51:  .ASCII\'RESPONSE'\
    170 003203      000                          .BYTE   0
```

```
    171 003204      042     127     110  SER52:  .ASCII\'WHEN A CONTINUE OR END FRAME DID NOT FOLLOW A START FRAME'N\
    172 003242      000                          .BYTE   0
    173 003243      042     127     110  SER53:  .ASCII\'WHEN AN END FRAME WAS SENT WITH NO START FRAME'N\
    174 003273      000                          .BYTE   0
    175 003274      042     127     110  SER54:  .ASCII\'WHEN AN END FRAME WITH A BAD CHECKSUM WAS SENT'N\
    176 003324      000                          .BYTE   0
    177 003325      042     127     110  SER55:  .ASCII\'WHEN A CONTINUE FRAME WAS SENT WITH NO START FRAME'N\
    178 003357      000                          .BYTE   0
    179 003360      042     127     110  SER56:  .ASCII\'WHEN TWO CONSECUTIVE START FRAMES WERE SENT'N\
    180 003407      000                          .BYTE   0
    181 003410      042     127     110  SER57:  .ASCII\'WHEN AN END FRAME WAS SENT AFTER A START FRAME TIMED OUT'N\
    182 003445      000                          .BYTE   0
    183
    184         003446                   OVL.MS  = .
```

```
                                        .SBTTL  OVERLAY 1  =  SETUP ( AND START TESTING )
003446                                  DMOVLY  SU,OVRLPT
003446  012143                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
000001                          SETUP   =       1
                                        .SBTTL  INIT DRIVE AND LOOK AT DRIVE SIGNALS

                                ;START OF TEST CODE
                                ;
                                ;INPUTS:
                                ;       LUNIT - LOGICAL UNIT NUMBER OF DRIVE UNDER TEST
                                ;       SDI - SDI INTERCONNECT CODE FOR DRIVE

                                ;INITIALIZE THE DRIVE

005247  104302  002636          STRTST: MOV SDI,R2                      ;GET SDI SELECT CODE
005251  060011                          XFC DINIT                       ;INITIALIZE THE DRIVE

                                ;WAIT FOR DRIVE TO ASSERT RECEIVER READY
                                ;TIME OUT AFTER ...?

005252  104201  001400                  MOV     #1400,R1                ; SETUP TIMEOUT
005254  117401                  1$:     DEC     R1                      ; DECREMENT DELAY
005255  055254                          BNE     1$                      ; LOOP UNTIL DONE
005256  114005                          CLR     R5                      ; GET TIMEOUT COUNTER
005257  104201  000310          2$:     MOV     #200.,R1                ; INNER LOOP TIMEOUT
005261  117401                  3$:     DEC     R1                      ; DELAY
005262  055261                          BNE     3$                      ;  UNTIL DONE
005263  020720                          CALL    RTDS                    ; GET REAL TIME DRIVE STATE
005264  115002                          TST     R2                      ; SEE IF ERROR OCCURRED
005265  015271                          BEQ     4$                      ; IF OK, CONTINUE
005266                                  PUSH    R1
005266  100461                                                          MOV R1,-(SP)
005267  021732                          CALL    TESTEW                  ; IF SO, BRANCH
005270                                  POP     R1
005270  104261                                                          MOV (SP)+,R1
005271  102201  000001          4$:     BIT     #RCVRDY,R1              ; CHECK RECEIVER READY LINES
005273  055317                          BNE     T                       ; ADVANCE IF ASSERTED
005274  117405                          DEC     R5                      ; DECREMENT TIME OUT COUNTER
005275  055257                          BNE     2$                      ; IF UNEXPIRED, BRANCH
005276                                  HARDER  9                       ;REPORT ERROR
005276  104200  000306  001037                                                  MOV     #MS9,OUT.04
005301  104300  001475  001036                                                  MOV     LUNIT,OUT.03
005304  104202  105701                                                          MOV     #9!ERHARD+3000.,R2
005306  104020  001035                                                          MOV     R2,OUT.02
005310  104200  005310  001034                                                  MOV     #.,OUT.01
005313  104200  060013  001033                                                  MOV     #ERRMES,OUT.RQ
005316  021740                          CALL    TESTEV                  ;EXIT TESTING THIS DRIVE
```

GET DRIVE CHARACTERISTICS

```
                                        .SBTTL  GET DRIVE CHARACTERISTICS
                                ;GET DRIVE CHARACTERISTICS

005317  104200  000003  001461  T:    MOV     #3,SDISTO               ; SET UP TEMPORARY SHORT TIMEOUT VALUE
005322  104203  002657                MOV     #CR.GCR,R3              ; POINT TO GET CHARS COMMAND
005324  104200  002404  002637        MOV     #MS.GCR,COMND
005327  021076                        CALL    TALK                    ; SDI INTERCHANGE
005330  115002                        TST     R2                      ; SEE IF ERROR OCCURRED
005331  015333                        BEQ     1$                      ; IF NOT, BRANCH
005332  021746                        CALL    TESTEX                  ; IF SO, REPORT
005333  104307  002760          1$:   MOV     CR+SHRTTO,R0            ; GET SHORT TIMEOUT
005335  103207  177760                BIC     #LBLONB,R0              ; CLEAR UNUSED BITS
005337  021463                        CALL    TO                      ; SET UP TIMEOUT
005340  104070  001461                MOV     R0,SDISTO               ; SAVE IN SHORT TIMEOUT
005342  104307  002761                MOV     CR+LONGTO,R0            ; GET LONG TIMEOUT
005344  103207  177760                BIC     #LBLONB,R0              ; CLEAR UNUSED BITS
005346  021463                        CALL    TO                      ; SET UP TIMEOUT
005347  104070  001462                MOV     R0,SDILTO               ; SAVE IN LONG TIMEOUT



                                        .SBTTL  ISSUE ONLINE COMMAND
                                ;ISSUE ONLINE COMMAND AND CHECK RESPONSE

005351  104203  002640                MOV     #CR.ONL,R3              ; POINT TO ONLINE COMMAND
005353  104200  002360  002637        MOV     #MS.ONL,COMND           ; SET UP FOR ERROR
005356  021076                        CALL    TALK                    ; SDI INTERCHANGE
005357  115002                        TST     R2                      ; SEE IF ERROR OCCURRED
005360  015362                        BEQ     2$                      ; IF NOT, BRANCH
005361  021746                        CALL    TESTEX                  ; IF SO, REPORT
005362                          2$:
```

```
                                        .SBTTL  ISSUE ONLINE COMMAND
                                ;ISSUE GET STATUS COMMAND AND CHECK THAT ATTENTION DROPS
                                
005362  104203  002671                MOV     #CR.GST,R3              ; POINT TO GET STATUS COMMAND
005364  104200  002443  002637        MOV     #MS.GST,COMND           ; SET UP FOR ERROR
005367  021076                        CALL    TALK                    ; SDI INTERCHANGE
005370  115002                        TST     R2                      ; SEE IF ERROR OCCURRED
005371  015373                        BEQ     TOA                     ; IF NOT, BRANCH
005372  021746                        CALL    TESTEX                  ; IF SO, REPORT
005373  020720                TOA:    CALL    RTDS                    ; GET REAL TIME DRIVE STATE
005374  102201  000002                BIT     #ATTN,R1                ; IS ATTENTION ASSERTED?
005376  015400                        BEQ     TOB                     ; COMMAND SHOULD HAVE CLEARED IT
                                ;       ERROR - ATTEN NOT DEASSERTED
005377  021732                        CALL    TESTEW
005400                          TOB:
                                
                                
                                
                                
                                
                                        .SBTTL  ISSUE DRIVE CLEAR COMMAND
                                ;ISSUE DRIVE CLEAR COMMAND
                                
005400  104200  000377  002731        MOV     #LOBYTE,ERRORS          ; MOVE TO DRIVE CLEAR SDI AREA
005403  104203  002645                MOV     #CR.CLR,R3              ; POINT TO DRIVE CLEAR
005405  104200  002365  002637        MOV     #MS.CLR,COMND           ; SET UP FOR ERROR
005410  021076                        CALL    TALK                    ; SDI INTERCHANGE
005411  115002                        TST     R2                      ; SEE IF ERROR OCCURRED
005412  015414                        BEQ     1$                      ; IF NOT, BRANCH
005413  021746                        CALL    TESTEX                  ; IF SO, REPORT
```

ISSUE CHANGE MODE COMMAND

```
                                                .SBTTL  ISSUE CHANGE MODE COMMAND
                                        ;       ISSUE CHANGE MODE COMMAND TO ENABLE DIAG CYL ACCESS
005414                                  1$:
005414  104203  002676                        MOV     #CR.MOD,R3              ; POINT TO CHANGE MODE COMMAND
005416  104300  002746  002745                MOV     MODE1,MODE              ; WRITE PROTECT, DIAG CYL ACCESS, NO FORMATTING
005421  104200  002452  002637                MOV     #MS.MOD,COMND           ; SET UP FOR ERROR
005424  021076                                CALL    TALK                    ; SDI INTERCHANGE
005425  115002                                TST     R2                      ; SEE IF ERROR OCCURRED
005426  015430                                BEQ     2$                      ; IF NOT, BRANCH
005427  021746                                CALL    TESTEX                  ; IF SO, REPOR
                                                .SBTTL  SPIN UP DRIVE
                                        ;SPIN UP DRIVE
005430  104203  002715                  2$:   MOV     #CR.RUN,R3              ; POINT TO RUN COMMAND
005432  104200  002502  002637                MOV     #MS.RUN,COMND           ; SET UP FOR ERROR
005435  021076                                CALL    TALK                    ; SDI INTERCHANGE
005436  115002                                TST     R2                      ; SEE IF ERROR OCCURRED
005437  015441                                BEQ     3$                      ; IF NOT, BRANCH
005440  021746                                CALL    TESTEX                  ; IF SO, REPORT
005441                                  3$:
```

```
                                        .SBTTL  ISSUE INITIATE RECALIBRATE COMMAND
                                ; ISSUE INITIATE RECALIBRATE COMMAND
                                
005441  104203  002710                  MOV     #CR.INR,R3              ; POINT TO RECALIBRATE COMMAND
005443  104200  002466  002637          MOV     #MS.INR,COMND           ; SET UP FOR ERROR
005446  021076                          CALL    TALK                    ; SDI INTERCHANGE
005447  115002                          TST     R2                      ; SEE IF ERROR OCCURRED
005450  015454                          BEQ     4$                      ; IF NOT, BRANCH
005451                                  PUSH    R1
005451  100461                                                                  MOV R1,-(SP)
005452  021746                          CALL    TESTEX
005453                                  POP     R1
005453  104261                                                                  MOV (SP)+,R1
005454  104204  000004          4$:     MOV     #4,R4                   ; R4 IS DECREMENT COUNTER
005456  020720                  T3C:    CALL    RTDS                    ; GET DRIVE SIGNALS
005457  115002                          TST     R2                      ; SEE IF ERROR
005460  051732                          BNE     TESTEW                  ; IF SO, BRANCH
005461  102201  000002                  BIT     #ATTN,R1                ; DID ATTENTION SET?
005463  015562                          BEQ T3D                         ;  NO
                                ; CHECK ERROR BITS - IF SET, REPORT "AFTER RECAL ERROR BITS ARE SET" OR SOMETHING THEN EXIT
                                ; IF LOGGABLE INFO BIT SET (NO ERROR BITS) MESSAGE- LOGGABLE INFO AFTER RECAL, CONTINUE TEST
005464                                  PUSH    R1                      ; SAVE R1
005464  100461                                                                  MOV R1,-(SP)
005465  104203  002671                  MOV     #CR.GST,R3              ; POINT TO GET STATUS COMMAND
005467  104200  002443  002637          MOV     #MS.GST,COMND           ; SET UP FOR ERROR
005472  021076                          CALL    TALK                    ; SEND AND RECEIVE COMMAND
005473  115002                          TST     R2                      ; ANY ERRORS?
005474  055525                          BNE     5$                      ; IF SO, EXIT
005475  104303  002753                  MOV     ST+ST.ERR,R3            ; GET ERROR INFORMATION
005477  102203  000350                  BIT     #<ST.FE+ST.RE+ST.PE+ST.WE>,R3 ; ANY ERRORS?
005501  015532                          BEQ     T3A                     ; IF OK CONTINUE
005502                                  HARDER  20,#SER22
005502  104200  001050  001037                  MOV     #MS20,OUT.04
005505  104200  003052  001040                  MOV     #SER22,OUT.05
005510  104300  001475  001036                  MOV     LUNIT,OUT.03
005513  104202  105714                          MOV     #20!ERHARD+3000.,R2
005515  104020  001035                          MOV     R2,OUT.02
005517  104200  005517  001034                  MOV     #.,OUT.01
005522  104200  060013  001033                  MOV     #ERRMES,OUT.RQ
005525                          5$:     POP     R1
005525  104261                                                                  MOV (SP)+,R1
005526  104203  001041                  MOV     #OUT.06,R3              ; ELSE SET UP ERROR INFO
005530  021752                          CALL    TESTEY
005531  005610                          BR      T4A
005532  102200  000010  002752  T3A:    BIT     #ST.EL,ST+1             ; ANY LOGGABLE INFO?
005535  015561                          BEQ     T3B                     ; IF NOT CONTINUE
005536  104203  001037                  MOV     #OUT.04,R3              ; IF SO, SET UP POINTER
005540  022131                          CALL    STRST                   ; AND SET UP INFO
005541                                  MSSG    21,#SER22
005541  104200  001073  001035                  MOV     #MS21,OUT.02
005544  104200  003052  001036                  MOV     #SER22,OUT.03
005547  100467                                                                  MOV R0,-(SP)
005550  100461                                                                  MOV R1,-(SP)
005551  104300  001475  001034                  MOV     LUNIT,OUT.01
005554  104207  060015                          MOV     #MESSAG,R0
005556  021000                                  CALL    HOSTRQ
005557  104261                                                                  MOV (SP)+,R1
```

```
                005560  104267
                                                                                MOV (SP)+,R0
 39             005561                          T3B:    POP     R1              ; RESTORE R1
                005561  104261
                                                                                MOV (SP)+,R1
 40             005562  102201  100000          T3D:    BIT     #RWRDY,R1       ; DID R/W READY SET?
 41             005564  055610                          BNE T4A                 ; IF SO, BRANCH
 42             005565  117404                          DEC     R4              ; TRY AGAIN?
 43             005566  055456                          BNE     T3C
 44             005567                                  HARDER  16              ;R/W READY DID NOT SET AFTER RECALIBRATE COMMAND
                005567  104200  000635  001037                  MOV     #MS16,OUT.04
                005572  104300  001475  001036                  MOV     LUNIT,OUT.03
                005575  104202  105710                          MOV     #16!ERHARD+3000.,R2
                005577  104020  001035                          MOV     R2,OUT.02
                005601  104200  005601  001034                  MOV     #.,OUT.01
                005604  104200  060013  001033                  MOV     #ERRMES,OUT.R0
 45             005607  021732                          CALL    TESTEW
```

```
                                                .SBTTL  GET SUBUNIT CHARACTERISTICS
                                        ;       GET SUBUNIT CHARACTERISTICS

005610  104301  002635                T4A:    MOV     UNITNB,R1               ; FIND SUBUNIT MASK FOR COMMAND
005612  103201  177774                        BIC     #^C3,R1                 ; CLEAR UNUSED BITS OF INDEX
005614  104207  000010                        MOV     #10,R0
005616  110207                        1$:     ROL     R0                      ; ROTATE
005617  117401                                DEC     R1                      ;  UNTIL DONE
005620  035616                                BPL     1$
005621  103207  177417                        BIC     #LBHINB,R0              ; CLEAR UNUSABLE BITS
005623  104070  002734                        MOV     R0,SUBUNT
005625  104203  002664                        MOV     #CR.SCR,R3              ; POINT TO GET SUBUNIT CHARACTERISTICS
005627  104200  002423  002637              MOV     #MS.SCR,COMND           ; SET UP FOR ERROR
005632  021076                                CALL    TALK                    ; SDI INTERCHANGE
005633  115002                                TSTB    R2                      ; SEE IF ERROR OCCURRED
005634  015636                                BEQ     2$
005635  021746                                CALL    TESTEX


                                                .SBTTL  SEEK TO CYLINDER 0, GROUP 0
                                        ;SEEK TO CYLINDER 0, GROUP 0

005636  104201  003063                2$:     MOV     #TSTCYL,R1
005640  114000  003063                        CLR     TSTCYL                  ; LO ORDER CYL 0
005642  104300  002774  003064              MOV     SUB+LBNCYL+1,TSTCYL+1   ; HI ORDER CYL 0
005645  103200  007777  003064              BIC     #HBHINB,TSTCYL+1
005650  114000  003065                        CLR     TSTCYL+2                ; GROUP 0
005652  022156                                CALL SEEK
```

```
                                        .SBTTL  CALCULATE NUMBERS
                                ;       CALCULATE NUMBER OF SECTORS PER TRACK
                                ;       NO. OF SECTORS/TRACK =(NO. OF LBN'S/TRACK) + (NO. OF RBN'S/TRACK)

005653  104301  002777        T4A1:   MOV SUB+RBNTRK,R1               ; GET RBN'S/TRACK
005655  103201  177600                BIC #177600,R1
005657  104307  003004                MOV SUB+LBNTRK,R0               ; GET LBN'S/TRACK
005661  103207  177400                BIC #HIBYTE,R0                  ; ZERO UPPER BITS
005663  105071                        ADD R0,R1                       ; ADD NUMBER OF LBN'S TO RBN'S
005664  104010  003060                MOV R1,SECTRK                   ; SAVE
005666  105011                        ADD R1,R1                       ; COMPUTE SECTORS TIMES TWO
005667  104010  003020                MOV R1,NSCSL                    ; AS SECTORS READ BEFORE DECLARING
                                                                      ; NO HEADER FOUND

                                ;COMPUTE FIRST CYLINDER IN XBN AREA

005671  104301  002773                MOV     SUB+LBNCYL,R1
005673  104010  003056                MOV     R1,FXBNCYL              ; FIRST XBN CYLINDER (LO)
005675  104307  002774                MOV     SUB+LBNCYL+1,R0
005677  103207  170000                BIC     #^CHBHINB,R0            ;CLEAR SUBUNIT CYL BITS
005701  104070  003057                MOV     R0,FXBNCYL+1            ; FIRST XBN CYLINDER (HI)

                                ;COMPUTE FIRST CYLINDER IN DBN AREA

005703  105301  003014                ADD SUB+XBNCYL,R1               ; ADD NUMBER OF XBN CYLINDERS
005705  045707                        BCC T4B                         ; BRANCH IF NO CARRY
005706  115407                        INC R0                          ; INCREMENT HI ORDER CYL
005707  104010  003054        T4B:    MOV R1,FDIACYL                  ; STORE STARTING DBN CYLINDER
005711  104070  003055                MOV R0,FDIACYL+1

                                ;COMPUTE LAST CYLINDER IN DBN AREA

005713  104303  003015                MOV     SUB+DBNCYL,R3           ; GET NUMBER OF DBN CYLINDERS
005715  110703                        SWAB    R3
005716  103203  177400                BIC     #HIBYTE,R3
005720  055742                        BNE     T4C
005721                                HARDER  17                      ;CHARACTERISTICS SAY LESS THAN 1 DIAGNOSTIC CYLINDER
005721  104200  000677  001037                                                MOV     #MS17,OUT.04
005724  104300  001475  001036                                                MOV     LUNIT,OUT.03
005727  104202  105711                                                        MOV     #17!ERHARD+3000.,R2
005731  104020  001035                                                        MOV     R2,OUT.02
005733  104200  005733  001034                                                MOV     #.,OUT.01
005736  104200  060013  001033                                                MOV     #ERRMES,OUT.RQ
005741  021740                        CALL    TESTEV
005742                          T4C:
                                ; NO LONGER NEED LAST DIAG CYL - JUST SAVE STARTING DBN, LAST TRACK, LAST GROUP, LAST CYL
                                ;               AND USE COMPUT XFC HEAVILY
005742  117403                        DEC     R3                      ; REDUCE BY ONE AND ADD TO
005743  105031                        ADD     R3,R1                   ; FIRST DBN CYLINDER TO
005744  045746                        BCC     T4D                     ; COMPUTE LAST CYLINDER
005745  115407                        INC     R0
005746  104010  003051        T4D:    MOV     R1,LDIACYL              ; STORE LAST DBN CYLINDER
005750  104070  003052                MOV     R0,LDIACYL+1
005752  104303  002775                MOV     SUB+GRPCYL,R3           ; GET GROUPS/CYLINDER
005754  103203  177400                BIC     #HIBYTE,R3              ; CLEAR UNUSED BITS
005756  117403                        DEC     R3                      ; LAST GROUP
005757  104030  003053                MOV     R3,LDIACYL+2
```

```
                                        .SBTTL  READ ALL FACTORY FORMATTED TRACKS
                                ;READ ALL FACTORY FORMATTED TRACKS.
                                ;REPORT ERR ? IF ALL SECTORS ON A TRACK READ WITH AN ERROR.
                                
                                ;COMPUTE DL . OF FIRST BLOCK ON FACTORY FORMATTED CYLINDER
                                
005761  114005                  T6:     CLR     R5
005762  104303  002776                  MOV     SUB+TRKGRP,R3           ; TRACKS/GROUP
005764  103203  177400                  BIC     #HIBYTE,R3
005766  104304  003060                  MOV     SECTRK,R4               ; X SECTORS/TRACK
005770  105035                  T6A:    ADD     R3,R5                   ; TO COMPUTE NUMBER OF SECTORS PER GROUP
005771  117404                          DEC     R4
005772  055770                          BNE     T6A
005773  104050  003073                  MOV     R5,SECGRP               ; SAVE SECTORS/GROUP
005775                                  PUSH    R2                      ; SAVE R2
005775  100462                                                          MOV R2,-(SP)
005776  104302  002775                  MOV     SUB+GRPCYL,R2           ; GET GROUPS/CYL
006000  103202  177400                  BIC     #HIBYTE,R2              ; CLEAR UNUSED BITS
006002                                  PUSH    R2                      ; SAVE R2
006002  100462                                                          MOV R2,-(SP)
006003  114000  003074                  CLR     SECCYL
006005  105300  003073  003074  2$:     ADD     SECGRP,SECCYL           ; GET SECTORS/CYL
006010  117402                          DEC     R2
006011  056005                          BNE     2$
006012                                  POP     R2                      ; RESTORE R2
006012  104262                                                          MOV (SP)+,R2
006013  104301  003015                  MOV     SUB+DBNCYL,R1           ; GET CYLINDERS IN DBN AREA
006015  110701                          SWAB    R1
006016  103201  177400                  BIC     #HIBYTE,R1              ; TO COMPUTE BLOCK NUMBER OF FIRST BLOCK
006020  114005                          CLR     R5                      ; CLEAR PRODUCT
006021  105025                  1$:     ADD     R2,R5
006022  117401                          DEC     R1
006023  056021                          BNE     1$
006024                                  POP     R2                      ; RESTORE R2
006024  104262                                                          MOV (SP)+,R2
006025  114007                          CLR     R0
006026                                  PUSH    R5                      ; SAVE NUMBER OF GROUPS IN DIAGNOSTIC AREA
006026  100465                                                          MOV R5,-(SP)
006027  117405                          DEC     R5                      ; SO DBN COMPUTED IS FIRST ON LAST GROUP
006030  105301  003073          T6C:    ADD     SECGRP,R1
006032  046034                          BCC     T6D
006033  115407                          INC     R0
006034  117405                  T6D:    DEC     R5
006035  056030                          BNE     T6C
006036  104010  003047                  MOV     R1,LCDBN
006040  104070  003050                  MOV     R0,LCDBN+1
006042                                  POP     R5                      ; R5 IS NUMBER OF GROUPS IN DIAG AREA
006042  104265                                                          MOV (SP)+,R5
006043  104307  003015                  MOV     SUB+DBNCYL,R0           ; GET NUMBER OF READ ONLY GROUPS
006045  103207  177400                  BIC     #HIBYTE,R0              ; CLEAR UNUSED BITS
006047  056071                          BNE     1$                      ; IF READ ONLY GROUPS > 0 THEN BRANCH
006050                                  HARDER  14                      ; ZERO READ ONLY GROUPS
006050  104200  000501  001037                  MOV     #MS14,OUT.04
006053  104300  001475  001036                  MOV     LUNIT,OUT.03
006056  104202  105706                          MOV     #14!ERHARD+3000.,R2
006060  104020  001035                          MOV     R2,OUT.02
006062  104200  006062  001034                  MOV     #.,OUT.01
```

```
                006065  104200  060013  001033
                                                                        MOV     #ERRMES,OUT.RQ
     47 006070  001740                          BR      TESTEV
     48 006071  104300  003051  003044  1$:     MOV     LDIACYL,ROFDC
     49 006074  104300  003052  003045          MOV     LDIACYL+1,ROFDC+1
     50 006077  107075                          SUB     R0,R5           ; R5 HAS NUMBER OF READ/WRITE GROUPS
     51 006100  016102                          BEQ     2$              ; IF R/W GROUPS ZERO, ERROR
     52 006101  036123                          BPL     3$              ; IF GREATER THAN ZERO, BRANCH
     53 006102                          2$:     HARDER  15              ; NO READ/WRITE GROUPS
        006102  104200  000555  001037                                          MOV     #MS15,OUT.04
        006105  104300  001475  001036                                          MOV     LUNIT,OUT.03
        006110  104202  105707                                                  MOV     #15!ERHARD+3000.,R2
        006112  104020  001035                                                  MOV     R2,OUT.02
        006114  104200  006114  001034                                          MOV     #.,OUT.01
        006117  104200  060013  001033                                          MOV     #ERRMES,OUT.RQ
     54 006122  001740                          BR      TESTEV
     55 006123  104304  002775          3$:     MOV     SUB+GRPCYL,R4   ; GET NUMBER OF GROUPS/CYL
     56 006125  103204  177400                  BIC     #HIBYTE,R4      ; CLEAR UNUSED BITS
     57 006127  107047                  4$:     SUB     R4,R0           ; SUBTRACT NUMBER OF GROUPS/CYL
     58 006130  076141                          BMI     5$
     59 006131  016146                          BEQ     6$
     60 006132  107200  000001  003044          SUB     #1,ROFDC        ; DECREMENT STARTING READ ONLY CYL BY 1
     61 006135  046127                          BCC     4$              ; IF NO CARRY, LOOP
     62 006136  117400  003045                  DEC     ROFDC+1         ; PROPOGATE BORROW
     63 006140  006127                          BR      4$              ; BRANCH
     64 006141  104203  177777          5$:     MOV     #-1,R3          ; SET UP FOR COMPLEMENT
     65 006143  103073                          BIC     R0,R3           ; COMPLEMENT (1'S)
     66 006144  115403                          INC     R3              ; 2'S COMPLEMENT
     67 006145  104037                          MOV     R3,R0           ; RESTORE TO R0
     68 006146  104070  003046          6$:     MOV     R0,ROFDC+2      ; SAVE IN STARTING GROUP
     69 006150  114003                          CLR     R3              ; CLEAR LO ORDER FIRST READ ONLY DBN
     70 006151  114004                          CLR     R4              ; CLEAR HI ORDER FIRST READ ONLY DBN
     71 006152  105303  003073          7$:     ADD     SECGRP,R3       ; ADD SEC/GRP TO LO ORDER STRT SEC
     72 006154  046156                          BCC     8$              ; IF NO CARRY, BRANCH
     73 006155  115404                          INC     R4              ; PROPOGATE CARRY
     74 006156  117405                  8$:     DEC     R5              ; DECREMENT COUNT
     75 006157  056152                          BNE     7$              ; LOOP
     76 006160  104030  003042                  MOV     R3,ROFDBN
     77 006162  104040  003043                  MOV     R4,ROFDBN+1
     78 006164  104030  003071                  MOV     R3,CURBLK
     79 006166  104040  003072                  MOV     R4,CURBLK+1
     80 006170  104200  000104  003102          MOV     #'D,LETTER      ; MAKE DBN'S
     81                                 ; POINT TO CURBLK, CALL SOME SUB THAT LOOKS AT THE POINTER PASSED
     82                                 ; AND THE 'LETTER' AND COMPUTS CYL, TRK, AND GRP. THEN RETURN
     83                                 ; THEN SEE IF CYL COMPUTED IS > MAX CYL IF SO, THIS PHASE OF TEST IS OVER (BR TGX)
     84 006173  104205  003071          T6G:    MOV     #CURBLK,R5      ;
     85 006175  021776                          CALL    FNDCY2          ; GO DIRECTLY TO DBN WILL BE PROCESSED
     86                                 ; THEN SEE IF CYL COMPUTED IS > MAX CYL. IF SO, THIS PHASE OF TEST IS OVER (BR TGX)
     87 006176  106300  003052  003064  1$:     CMP     LDIACYL+1,TSTCYL+1
     88 006201                                  BCS     T6F
        006201  046203                                                          BCC     .+2
        006202  006233                                                          BR      T6F
     89 006203  106300  003051  003063          CMP     LDIACYL,TSTCYL
     90 006206                                  BCS     T6F
        006206  046210                                                          BCC     .+2
        006207  006233                                                          BR      T6F
     91 006210  104201  003063                  MOV     #TSTCYL,R1
     92 006212  022156                          CALL    SEEK            ; SEEK TO REQUESTED CYL
```

```
     93 006213  104205  003071                  MOV     #CURBLK,R5
     94 006215  104303  003060                  MOV     SECTRK,R3
     95 006217  022325                  T6E:    CALL    READ1                   ;READ EACH SECTOR TILL ONE READS OK
     96 006220  103200  177400  003072          BIC     #HIBYTE,CURBLK+1
     97 006223  105300  003060  003071          ADD     SECTRK,CURBLK
     98 006226  046173                          BCC     T6G
     99 006227  115400  003072                  INC     CURBLK+1
    100 006231  115404                          INC     R4
    101 006232  006173                          BR      T6G
    102
    103 006233  001633                  T6F:    BR      OVRLAY                  ; BRING IN REST OF THE CODE
    104
    105         006232                  OVL.SU  =       .-TEST
```

```
                                        .SBTTL  SEEK TO SELECTED GROUP, READ HDR, SEEK TO DIAGNOSTIC AREA
                                        DMOVLY  TS,OVRLPT
006234  167077                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
000002                          TEST    =       2
                                ;STARTING WITH CYLINDER 0, GROUP 0 AND INCREMENTING THROUGH EVERY GROUP
                                ;ON THE DISK, PERFORM A SEEK TO THE SELECTED GROUP, READ A HEADER
                                ;ON THAT GROUP AND THEN A SEEK TO THE FACTORY FORMATTED DIAGNOSTIC
                                ;GROUP TO VERIFY HEADS POSITIONED CORRECTLY.

                                ;COMPUTE LBNS PER GROUP

005247  114001                  T7:     CLR     R1                      ; LBN'S/TRK X TRK/GROUP
005250  104305  002776                  MOV     SUB+TRKGRP,R5
005252  103205  177400                  BIC     #HIBYTE,R5
005254  104303  003004                  MOV     SUB+LBNTRK,R3           ;GET LBN'S PER TRACK
005256  103203  177400                  BIC     #HIBYTE,R3              ;CLEAR UNUSED BITS
005260  104030  003066                  MOV     R3,BLOCKT
005262  105031                  T7A:    ADD     R3,R1
005263  117405                          DEC     R5
005264  055262                          BNE     T7A
005265  104010  003067                  MOV     R1,BLOCKG               ;SAVE IN BLOCKG
005267  114000  003070                  CLR     BLOCKC
005271  104307  002775                  MOV     SUB+GRPCYL,R0
005273  103207  177400                  BIC     #HIBYTE,R0
005275  105300  003067  003070  1$:     ADD     BLOCKG,BLOCKC
005300  117407                          DEC     R0
005301  055275                          BNE     1$
005302  114000  003063                  CLR     TSTCYL
005304  114000  003065                  CLR     TSTCYL+2                ;GROUP ZERO
005306  114000  003061                  CLR     TSTBLK
005310  114000  003062                  CLR     TSTBLK+1
005312  114000  003106                  CLR     GRPCNT
005314  114000  003107                  CLR     OLDGRP                  ; CLEAR OLD GROUP VALUE USED IN FNDCYL
005316  104200  000114  003101          MOV     #'L,AREA                ;AREA BEING SEEKED INTO IS LBNS
005321  104300  003101  003102  T7C:    MOV     AREA,LETTER             ; MOVE TO LETTER FOR REPORTING
005324  104205  003061                  MOV     #TSTBLK,R5              ;POINT TO LBN NUMBER FOR -> VAR1
                                ; COMPUTE CYL, TRK AND GRP.
                                ; IF CYL < STARTING CYL THEN CONTINUE ELSE IF CYL >= STARTING DIAG CYL THEN OUT (BR T7X)
                                ; ELSE IF CYL = STARTING XBN CYL CLR TSTBLK, TSTBLK+1 AND BR T7C
005326  021770                          CALL    FNDCYL                  ;FIND CYLINDER
005327  106300  003056  003116          CMP     FXBNCYL,CYLLO           ;IS CYL = STARTING XBN?
005332  055366                          BNE     1$                      ;IF NOT, BRANCH
005333  106300  003057  003117          CMP     FXBNCYL+1,CYLLO+1       ;
005336  055363                          BNE     1$
005337  106200  000130  003101          CMP     #'X,AREA                ; IF HERE AGAIN, BRANCH
005342  015366                          BEQ     1$
005343  104200  000130  003101          MOV     #'X,AREA                ; NOW IN XBN AREA
005346  114000  003061                  CLR     TSTBLK                  ;IF SO, CLEAR TSTBLK
005350  114000  003062                  CLR     TSTBLK+1
005352  104300  003060  003066          MOV     SECTRK,BLOCKT           ;CHANGE BLOCK COUNT PER TRACK
005355  104300  003073  003067          MOV     SECGRP,BLOCKG           ;CHANGE BLOCK COUNT PER GROUP
005360  104300  003074  003070          MOV     SECCYL,BLOCKC           ;CHANGE BLOCK COUNT PER CYL
005363  114000  003107                  CLR     OLDGRP                  ; CLEAR OLD GROUP
```

```
     57 005365  005321                                BR      T7C                     ; AND TRY AGAIN WITH XBN
     58 005366  106300  003054  003063  1$:     CMP     FDIACYL,TSTCYL          ;DONE?
     59 005371  055376                          BNE     2$                      ;IF NOT, CONTINUE
     60 005372  106300  003055  003064          CMP     FDIACYL+1,TSTCYL+1      ;DONE?
     61 005375  015470                          BEQ     T7X                     ;IF SO, EXIT
     62 005376  104300  003101  003102  2$:     MOV     AREA,LETTER
     63 005401  104201  003063                  MOV     #TSTCYL,R1              ;SEEK TO CYLINDER
     64 005403  022156                          CALL    SEEK
     65 005404  104205  003061                  MOV     #TSTBLK,R5              ; SET UP POINTER TO TEST BLOCK
     66 005406  104303  003060                  MOV     SECTRK,R3               ;GET MAXIMUM NUMBER OF SECTORS TO READ
     67 005410  022325                          CALL    READ1                   ;READ UNTIL AT LEAST ONE RECORD READ WITHOUT ERROR
     68                                                 ;
                                                        COMPUTE CYL FROM LCDBN
     69 005411  104200  000104  003102  3$:     MOV     #'D,LETTER              ; IN DIAGNOSTIC AREA
     70 005414  021770                          CALL    FNDCYL
     71 005415  104201  003063                  MOV     #TSTCYL,R1
     72 005417  104303  003060                  MOV     SECTRK,R3               ; R3 = SECTORS/TRACK
     73 005421  022156                          CALL    SEEK
     74 005422  104205  003047                  MOV     #LCDBN,R5               ; SET UP POINTER TO TEST BLOCK
     75 005424  022325                          CALL    READ1                   ;READ UNTIL AT LEAST ONE RECORD READ WITHOUT ERROR
     76 005425  115400  003106                  INC     GRPCNT
     77 005427  103200  177400  003106          BIC     #HIBYTE,GRPCNT
     78 005432  055453                          BNE     5$
     79 005433                                  PUSH    R0                      ; SAVE R0
        005433  100467                                                                 MOV R0,-(SP)
     80 005434  104207  060007                  MOV     #T4SOFT,R0              ; TO INTERRUPT HOST <<ONLY>>
     81 005436  104300  001475  001034          MOV     LUNIT,OUT.01            ; IDENTIFY UNIT
     82 005441  114000  001035                  CLR     OUT.02                  ; NO SOFT ERRORS
     83 005443  114000  001036                  CLR     OUT.03                  ; NO ECC CORRECTIONS
     84 005445  114000  001037                  CLR     OUT.04
     85 005447  114000  001040                  CLR     OUT.05
     86 005451  021000                          CALL    HOSTRQ                  ; REPORT (NOTHING)
     87 005452                                  POP     R0                      ; RESTORE R0
        005452  104267                                                                 MOV (SP)+,R0
     88 005453  105300  003070  003061  5$:     ADD     BLOCKC,TSTCLK
     89 005456  045461                          BCC     6$
     90 005457  115400  003062                  INC     TSTBLK+1
     91 005461  105300  003067  003061  6$:     ADD     BLOCKG,TSTBLK
     92 005464  045467                          BCC     7$
     93 005465  115400  003062                  INC     TSTBLK+1
     94 005467  005321                  7$:     BR      T7C
```

CHECK WRITE PROTECT

```
      1                                                  .SBTTL  CHECK WRITE PROTECT
      2 005470  104200  177777  003104  T7X:    MOV     #-1,WFLAG               ; SET FLAG
      3 005473  114000  003071                  CLR     CURBLK                  ; CLEAR BLOCK
      4 005475  114000  003072                  CLR     CURBLK+1                ;
      5 005477  114000  003076                  CLR     CURTRK                  ; CLEAR TRACK
      6 005501  104300  003054  003063          MOV     FDIACYL,TSTCYL          ; POINT TO CYL TO TEST
      7 005504  104300  003055  003064          MOV     FDIACYL+1,TSTCYL+1
      8 005507  114000  003065                  CLR     TSTCYL+2                ; CLEAR GROUP
      9 005511  104201  003063                  MOV     #TSTCYL,R1              ; R1 -> TEST CYL
     10 005513  022156                          CALL    SEEK                    ; SEEK THERE
     11 005514  026734                          CALL    WRITEB                  ; AND ATTEMPT TO WRITE
     12 005515  115007                          TST     R0                      ; WAS IT SUCCESSFUL?
     13 005516  055542                          BNE     1$                      ; IF NOT, THAT WAS EXPECTED
     14 005517  115001                          TST     R1                      ;
     15 005520  055542                          BNE     1$                      ;
     16 005521                                  HARDER  35                      ; REPORT ERROR
        005521  104200  002245  001037                          MOV     #MS35,OUT.04
        005524  104300  001475  001036                          MOV     LUNIT,OUT.03
        005527  104202  105733                                  MOV     #35!ERHARD+3000.,R2
        005531  104020  001035                                  MOV     R2,OUT.02
        005533  104200  005533  001034                          MOV     #.,OUT.01
        005536  104200  060013  001033                          MOV     #ERRMES,OUT.RQ
     17 005541  021740                          CALL    TESTEV
     18 005542  021761                  1$:     CALL    DR.CLR                  ; CLEAR DRIVE
     19 005543  114000  003104                  CLR     WFLAG                   ; CLEAR FLAG
```

```
                                                .SBTTL  CHANGE MODE TO ALLOW FORMATTING AND WRITING
005545                                  T8:
                                        ;
                                        ;       FIRST CHANGE MODE TO ALLOW FORMATTING AND WRITING
                                        ;
005545  104204  000004                  MOV     #4,R4                   ; R4 HAS DECREMENT COUNTER
005547  104203  002676          2$:     MOV     #CR.MOD,R3              ; POINT TO CHANGE MODE COMMAND
005551  104300  002747  002745          MOV     MODE2,MODE              ; WRITE PROTECT, DIAG CYL ACCESS, NO FORMATTING
005554  104200  002452  002637          MOV     #MS.MOD,COMND           ; SET UP FOR ERROR
005557  021076                          CALL    TALK                    ; INITIATE SDI INTERCHANGE
005560  115002                          TST     R2                      ; SEE IF ERROR OCCURRED
005561  015566                          BEQ     1$                      ; IF NOT, BRANCH
005562  021746                          CALL    TESTEX                  ; IF SO, ERROR
005563  117404                          DEC     R4                      ; DONE?
005564  055547                          BNE     2$                      ; IF NOT, BRANCH (TRY AGAIN BECAUSE FORMAT
                                        ;                               ;       WON'T WORK WITH A WRITE PROTECTED DRIVE)
005565  001617                          BR      TESTX                   ; ELSE, EXIT
005566  114000  003105          1$:     CLR     FLAG                    ; USE FLAG TO DECIDE IF WE FORMAT OR TEST A TRACK
005570                                  T08:
                                                .SBTTL  FORMAT A TRACK THEN CHECK IT.
                                        ;
                                        ;       FORMAT ALL TRACKS, THEN DO SEVERAL WRITES, READS AND DATA COMPARES
                                        ;       UNTIL ONE SECTOR PASSES.  DO THIS FOR EACH TRACK IN EACH GROUP
                                        ;       ON EACH CYLINDER IN THE DIAGNOSTIC AREA EXCEPT FOR THE LAST
                                        ;       (FACTORY FORMATTED) CYLINDER ON THE PACK
                                        ;
005570  104300  003054  003063          MOV     FDIACYL,TSTCYL          ; START AT FIRST DIAGNOSTIC CYL
005573  104300  003055  003064          MOV     FDIACYL+1,TSTCYL+1
005576  104201  003063                  MOV     #TSTCYL,R1              ; POINT TO CYLINDER TO SEEK TO
005600  114000  003065                  CLR     TSTCYL+2                ; START WITH GROUP 0
005602  022156                          CALL    SEEK                    ; SEEK TO FIRST DIAGNOSTIC CYLINDER
005603  114004                          CLR     R4                      ; START WITH DBN 0
005604  104040  003061                  MOV     R4,TSTBLK
005606  104040  003062                  MOV     R4,TSTBLK+1             ; CLEAR BLOCK
005610  104040  003076          T8A:    MOV     R4,CURTRK               ; SAVE TRACK NUMBER
005612  115000  003105                  TST     FLAG                    ; DO WE FORMAT
005614  055617                          BNE     1$                      ; IF NOT ZERO, TEST TRACK
005615  026405                          CALL    FORTRK                  ; FORMAT THE TRACK
005616  005620                          BR      2$                      ; DO NOT TEST TRACK UNTIL ALL FORMATTED
005617  026624                  1$:     CALL    TRKTST                  ; TEST THE TRACK
005620  105300  003060  003061  2$:     ADD     SECTRK,TSTBLK           ; ADD SECTORS/TRACK
005623  045626                          BCC     T8B                     ; IF NO CARRY, BRANCH
005624  115400  003062                  INC     TSTBLK+1                ; INCREMENT
005626  104304  003076          T8B:    MOV     CURTRK,R4               ; GET TRACK NUMBER
005630  115404                          INC     R4                      ; INCREMENT TRACK NUMBER
005631  104303  002776                  MOV     SUB+TRKGRP,R3           ; GET NUMBER OF TRACKS/GROUP
005633  103203  177400                  BIC     #HIBYTE,R3              ; CLEAR UNUSED BITS
005635  106034                          CMP     R3,R4                   ; SEE IF ALL TRACKS READ
005636  055610                          BNE     T8A                     ; IF NOT, BRANCH
005637  114004                          CLR     R4                      ; ZERO R4
005640  115400  003065                  INC     TSTCYL+2                ; MOVE TO NEXT GROUP
005642  104301  002775                  MOV     SUB+GRPCYL,R1           ; GET GROUPS/CYLINDER
005644  103201  177400                  BIC     #HIBYTE,R1              ; CLEAR UNUSED BITS
005646  106010  003065                  CMP     R1,TSTCYL+2             ; COMPARE
005650  055660                          BNE     T8D                     ; IF ALL GROUPS NOT TESTED, BRANCH
005651  104040  003065                  MOV     R4,TSTCYL+2             ; START WITH GROUP 0 ON NEW CYLINDER
005653  115400  003063                  INC     TSTCYL                  ; INCREMENT CYLINDER
```

```
                                                BCC     T8D                     ; IF NO CARRY, BRANCH
                                                INC     TSTCYL+1                ; INCREMENT HI CYLINDER
                                        T8D:    CMP     TSTCYL,ROFDC            ; SEE IF ON LAST CYLINDER (LO ORDER)
                                                BNE     1$                      ; IF NOT, BRANCH
                                                CMP     TSTCYL+1,ROFDC+1        ; SEE IF ON LAST CYLINDER (HI ORDER)
                                                BNE     1$                      ; IF NOT, BRANCH
                                                CMP     TSTCYL+2,ROFDC+2        ; SEE IF ON RESERVED GROUPS
                                                BEQ     T8T                     ; IF SO, END THIS PHASE OF TEST
                                        1$:     MOV     #TSTCYL,R1              ; POINT TO NEW CYLINDER OR GROUP
                                                CALL    SEEK                    ; ISSUE SEEK
                                                BR      T8A                     ; BRANCH
                                        T8T:    TST     FLAG                    ; ARE WE DONE?
                                                BNE     T8E                     ; IF SO, BRANCH
                                                MOV     #-1,FLAG                ; ELSE, CHANGE FLAG
                                                BR      T08                     ; AND CHECK TRACKS

                                        ; *** NOW SEND INVALID SELECT GROUP COMMAND AND EXPECT ERROR
                                        T8E:    MOV     #1,SDILTO               ; CHANGE LONG TIMEOUT
                                                MOV     #SUB+GRPCYL,R0          ; R0 HAS GROUP #
                                                BIC     #HIBYTE,R0              ; STRIP OFF UNUSED BITS
                                                CMP     #LOBYTE,R0              ; IS THIS UNIT USE MAX BITS?
                                                BEQ     T11                     ; IF SO, BRANCH (DON'T DO TEST)
                                                BIS     #SL.GRP,R0              ; SET SELECT GROUP CODE
                                                MOV     R0,RUN                  ; STORE IN RUN CODE
                                                CLR     CR.RUN+1                ; SET UP FOR LEVEL 1 EXCHANGE
                                                MOV     #CR.RUN,R3              ; R3 -> PACKET
                                                MOV     SDI,R2                  ; SEND MESSAGE
                                                MOV     (R3)+,R0                ; SET UP PARAMETERS
                                                MOV     (R3),R1                 ;
                                                XFC     SEND                    ; SEND COMMAND
                                                TST     R1                      ; IF FAIL, BRANCH
                                                BNE     T8F
                                                MOV     #CR.GST,R3              ; CHECK OUT STATUS
                                                CALL    TALK
                                                BIT     #<ST.PE+ST.RE+ST.FE+ST.WE>,ST.ERR+ST ;ERROR?
                                                BNE     T8F                     ; IF SO, BRANCH
                                                HARDER  33                      ; ELSE ERROR
                                                                                        MOV     #MS33,OUT.04
                                                                                        MOV     LUNIT,OUT.03
                                                                                        MOV     #33!ERHARD+3000.,R2
                                                                                        MOV     R2,OUT.02
                                                                                        MOV     #.,OUT.01
                                                                                        MOV     #ERRMES,OUT.RQ
                                                CALL    TESTEV
                                        T8F:    CALL    DR.CLR                  ; CLEAR DRIVE OF ERROR
```

```
                                                .SBTTL  SEND INVALID LEVEL 2 COMMANDS
005771  114000  002637          T11:    CLR     COMND                   ; CLEAR COMMAND CODE AS A FLAG
005773  114007                          CLR     R0                      ; CLEAR SDI COMMAND VALUE
005774  104200  000001  002716          MOV     #1,CR.RUN+1             ; RESET PROPER LENGTH
005777  104070  002736          1$:     MOV     R0,RUN                  ; SET CODE
006001                                  PUSH    R0                      ; SAVE R0
006001  100467                                                                  MOV R0,-(SP)
006002  026263                          CALL    TSTCMD                  ; AND TEST
006003                                  POP     R0                      ; RESTORE R0
006003  104267                                                                  MOV (SP)+,R0
006004  115007                          TST     R0                      ; TRY INVALID OPCODE ONCE
006005  056011                          BNE     2$                      ; EXIT WHEN TRIED ONCE WITH INVALID CODE
006006  105207  000400                  ADD     #400,R0                 ; ADD TO NEXT CODE
006010  055777                          BNE     1$                      ; IF NOT DONE, TEST AGAIN
006011  104200  000377  002736  2$:     MOV     #377,RUN                ; TRY WITH LOW BITS SET
006014  026263                          CALL    TSTCMD                  ;
006015  104300  002750  002736          MOV     RUNCMD,RUN              ; RESTORE RUN COMMAND
006020  104205  000100                  MOV     #64.,R5
006022  104050  002716                  MOV     R5,CR.RUN+1             ; STORE INVALID COMMAND LENGTH
006024  104204  003171                  MOV     #SER50,R4               ;
006026  026263                          CALL    TSTCMD
006027  114005                          CLR     R5                      ; TRY AGAIN WITH 0 COMMAND LENGTH BUT INVALID COMMAN
006030  104204  003176                  MOV     #SER51,R4
006032  104050  002674                  MOV     R5,CR.GST+3             ; DESTROY RESPONSE LENGTH
006034  104203  002671                  MOV     #CR.GST,R3
006036  026265                          CALL    TSTCM2
006037  104200  000007  002674          MOV     #7,CR.GST+3             ; RESTORE COMMAND LENGTH
```

```
                                        .SBTTL  SEND INVALID LEVEL 1 COMMANDS

                                ; *** SEND START THEN SELECT GROUP
006042  114000  002716                 CLR     CR.RUN+1
006044  104200  070400  002736         MOV     #MS.STR,RUN             ; SET UP COMMAND
006047  026250                         CALL    SNDLV1                  ; SEND LEVEL 1 START COMMAND
006050  104200  003204  002637         MOV     #SER52,COMND
006053  104200  107000  002736         MOV     #SL.GRP,RUN             ; MOVE SELECT GROUP CODE INTO COMMAND
006056  026253                         CALL    SNDL1A                  ; SEND NEXT FRAME
006057  104200  131000  002736         MOV     #MS.END,RUN             ; SET UP END FRAME
006062  026263                         CALL    TSTCMD                  ; CHECK IF CAUGHT

                                ; *** SEND END WITH NO START
006063  104200  131000  002736         MOV     #MS.END,RUN             ; SET UP END FRAME
006066  104200  003243  002637         MOV     #SER53,COMND
006071  026263                         CALL    TSTCMD                  ; CHECK IF CAUGHT

                                ; *** SEND START THEN END WITH BAD CHECKSUM
006072  104200  070400  002736         MOV     #MS.STR,RUN             ; SET UP START FRAME
006075  026250                         CALL    SNDLV1                  ; SEND FRAME
006076  104200  003274  002637         MOV     #SER54,COMND
006101  104200  131000  002736         MOV     #MS.END,RUN             ; SET UP END FRAME (ZERO CHECKSUM)
006104  026263                         CALL    TSTCMD                  ; CHECK IF CAUGHT

                                ; *** SEND CONTINUE WITH NO START
006105  104200  152000  002736         MOV     #MS.CNT,RUN             ; SET CONTINUE FRAME
006110  101200  000014  002736         BIS     #RUNLBC,RUN
006113  026250                         CALL    SNDLV1                  ; SEND IT
006114  104200  131000  002736         MOV     #MS.END,RUN             ; SET UP END FRAME
006117  104200  003325  002637         MOV     #SER55,COMND
006122  026263                         CALL    TSTCMD                  ; CHECK IF CAUGHT
```

ISSUE DISCONNECT COMMAND

```
                                                .SBTTL  ISSUE DISCONNECT COMMAND
006123  104203  002652          T9:     MOV     #CR.DIS,R3              ; POINT TO DISCONNECT COMMAND
006125  104200  002375  002637          MOV     #MS.DIS,COMND           ; SET UP FOR ERROR
006130  021076                          CALL    TALK                    ; INITIATE SDI INTERCHANGE
006131  115002                          TST     R2                      ; SEE IF ERROR OCCURRED
006132  016134                          BEQ     T10
006133  021746                          CALL    TESTEX                  ; IF SO, BRANCH

                                                .SBTTL  CHECK AVAIL FLAG
006134  114005          T10:    CLR     R5                      ; SET UP TIMEOUT COUNTER
006135  104302  002636          T10LOP: MOV     SDI,R2                  ; MOVE MASK TO R2
006137  020754                          CALL    RDSTAT                  ; GET DRIVE STATUS
006140  115002                          TST     R2                      ; WAS IT OK?
006141  016207                          BEQ     2$                      ; IF NO ERROR, BRANCH
006142  102201  000400                  BIT     #RCVERR,R1              ; SEE IF ANY ERRORS
006144  056166                          BNE     1$                      ; IF NOT, ERROR
006145                                  HARDER  11                      ; REPORT INVALID STATUS ERROR
006145  104200  000366  001037                          MOV     #MS11,OUT.04
006150  104300  001475  001036                          MOV     LUNIT,OUT.03
006153  104202  105703                                  MOV     #11!ERHARD+3000.,R2
006155  104020  001035                                  MOV     R2,OUT.02
006157  104200  006157  001034                          MOV     #.,OUT.01
006162  104200  060013  001033                          MOV     #ERRMES,OUT.RQ
006165  001732                          BR      TESTEW                  ; BRANCH TO DONE
006166                          1$:
006166                                  HARDER  12                      ; REPORT XMIT ERROR
006166  104200  000434  001037                          MOV     #MS12,OUT.04
006171  104300  001475  001036                          MOV     LUNIT,OUT.03
006174  104202  105704                                  MOV     #12!ERHARD+3000.,R2
006176  104020  001035                                  MOV     R2,OUT.02
006200  104200  006200  001034                          MOV     #.,OUT.01
006203  104200  060013  001033                          MOV     #ERRMES,OUT.RQ
006206  001732                          BR      TESTEW                  ; BRANCH TO DONE
006207                          2$:
006207  102201  000100                  BIT     #AVAIL,R1               ; SEE IF AVAILABLE IS ASSERTED
006211  056241                          BNE     T12                     ; IF SO, BRANCH TO LAST TEST
006212  117405                          DEC     R5                      ; DECREMENT TIMEOUT COUNT
006213  056135                          BNE     T10LOP                  ; IF UNEXPIRED, BRANCH
006214  103201  077674                  BIC     #077674,R1              ; CLEAR UNUSED BITS
006216                                  HARDER  29,R1                   ; REPORT ERROR
006216  104200  002007  001037                          MOV     #MS29,OUT.04
006221  104010  001040                                  MOV     R1,OUT.05
006223  104300  001475  001036                          MOV     LUNIT,OUT.03
006226  104202  105725                                  MOV     #29!ERHARD+3000.,R2
006230  104020  001035                                  MOV     R2,OUT.02
006232  104200  006232  001034                          MOV     #.,OUT.01
006235  104200  060013  001033                          MOV     #ERRMES,OUT.RQ
006240  021740                          CALL    TESTEV


006241                          T12:
006241  104300  002750  002736          MOV     RUNCMD,RUN              ; RESTORE RUN COMMAND
006244  104200  000001  002716          MOV     #1,CR.RUN+1             ; RESTORE COMMAND LENGTH
006247  001617                          BR      TESTX                   ; BRANCH TO TEST NEXT UNIT
```

```
                                        .SBTTL  SNDLV1 AND TSTCMD
                                ; *** SEND LEVEL 1 START COMMAND
                                ;       SEND THE START FRAME AND RETURN
006250  101200  000014  002736  SNDLV1: BIS     #RUNLBC,RUN             ; SET RUN COMMAND
006253  104203  002715          SNDL1A: MOV     #CR.RUN,R3              ; SEND COMMAND
006255  104302  002636                  MOV     SDI,R2                  ; R2 -> PORT
006257  104237                         MOV     (R3)+,R0                ; SET UP PARAMETERS
006260  104131                         MOV     (R3),R1
006261  060004                         XFC     SEND                    ; SEND COMMAND
006262  000000                         RETURN

                                ; *** TEST THE COMMAND
                                ;       INPUT   RUN HAS LEVEL 1 OR 2 COMMAND SET UP
                                ;               CR.RUN = 0 TO INITIATE LEVEL 1 TRANMITTION
                                ;                      = WHATEVER FOR LEVEL 2
                                ;       OUTPUT  WILL NOT RETURN IF ERROR OCCURED
                                ;
006263  104203  002715          TSTCMD: MOV     #CR.RUN,R3              ; SEND COMMAND
006265  021076                  TSTCM2: CALL    TALK
006266  115002                          TST     R2                      ; DID IT FAIL?
006267  016277                          BEQ     T11ER                   ; IF NOT, BRANCH TO REPORT AND EXIT
006270  104203  002671                  MOV     #CR.GST,R3              ; CHECK STATUS
006272  021076                          CALL    TALK                    ; GET STATUS
006273  102200  000350  002753          BIT     #<ST.PE+ST.RE+ST.FE+ST.WE>,ST+ST.ERR
006276  056403                          BNE     TSTCME                  ; BRANCH IF ERROR

                                ; *** DID NOT FAIL -> ERROR
006277  115000  002637          T11ER:  TST     COMND                   ; DID WE HAVE COMMAND SET
006301  056357                          BNE     T11X2                   ; IF SO, LEVEL 1 XMIT ERROR
006302  106300  002750  002736          CMP     RUNCMD,RUN              ; ELSE, WHAT TYPE LEVEL 2 XMIT ERROR?
006305  056333                          BNE     T11X1                   ; IF RUN NOT SET TO RUN COMMAND, BRANCH
006306                                  HARDER  31,<R4,R5>
006306  104200  002077  001037                          MOV     #MS31,OUT.04
006311  104040  001040                                  MOV     R4,OUT.05
006313  104050  001041                                  MOV     R5,OUT.06
006315  104300  001475  001036                          MOV     LUNIT,OUT.03
006320  104202  105727                                  MOV     #31!ERHARD+3000.,R2
006322  104020  001035                                  MOV     R2,OUT.02
006324  104200  006324  001034                          MOV     #.,OUT.01
006327  104200  060013  001033                          MOV     #ERRMES,OUT.RQ
006332  001753                  BR      TESTED
006333                  T11X1:  HARDER  30,RUN
006333  104200  002053  001037                          MOV     #MS30,OUT.04
006336  104300  002736  001040                          MOV     RUN,OUT.05
006341  104300  001475  001036                          MOV     LUNIT,OUT.03
006344  104202  105726                                  MOV     #30!ERHARD+3000.,R2
006346  104020  001035                                  MOV     R2,OUT.02
006350  104200  006350  001034                          MOV     #.,OUT.01
006353  104200  060013  001033                          MOV     #ERRMES,OUT.RQ
006356  001753                  BR      TESTED
006357                  T11X2:  HARDER  32,COMND
006357  104200  002130  001037                          MOV     #MS32,OUT.04
006362  104300  002637  001040                          MOV     COMND,OUT.05
006365  104300  001475  001036                          MOV     LUNIT,OUT.03
006370  104202  105730                                  MOV     #32!ERHARD+3000.,R2
006372  104020  001035                                  MOV     R2,OUT.02
006374  104200  006374  001034                          MOV     #.,OUT.01
006377  104200  060013  001033                          MOV     #ERRMES,OUT.RQ
```

```
     36 006402  001753                        BR      TESTED
     37 006403  021761              TSTCME:   CALL    DR.CLR          ; CLEAR ERROR
     38 006404  000000                        RETURN
```

```
                                                .SBTTL  FORTRK - FORMAT TRACK
006405                                  FORTRK:
                                        ;
                                        ;       FORMAT TRACK GIVEN BY CURTRK, STARTING WITH DBN GIVEN IN TSTBLK
                                        ;       THE DATA WRITTEN IS UNPREDICTABLE
                                        ;
006405  104307  003061                          MOV     TSTBLK,R0               ; GET LO STARTING DBN
006407  104301  002776                          MOV     SUB+HIDBN,R1            ; GET HIGH ORDER BITS OF STARTING DBN
006411  110601                                  ROR     R1                      ; MOVE TO CORRECT POSITION
006412  110601                                  ROR     R1
006413  110601                                  ROR     R1
006414  110601                                  ROR     R1
006415  103201  170377                          BIC     #HBLONB,R1              ; CLEAR UNUSED BITS
006417  101201  140000                          BIS     #HD.DBN,R1              ; SET HEADER CODE
006421  101301  003062                          BIS     TSTBLK+1,R1             ; GET HI STARTING DBN
006423  104203  003243                          MOV     #OBUFF,R3               ; POINT TO OUTPUT BUFFER
006425  104304  003060                          MOV     SECTRK,R4               ; R4 CONTAINS NUMBER OF SECTORS TO FORMAT
006427  104205  003652                          MOV     #FCHAIN,R5              ; R5 POINTS TO FORMAT CHAIN
006431  100253                          FLOOP:  MOV     R3,(R5)+                ; MOVE POINTER TO BUFFER TO CHAIN
006432  100257                                  MOV     R0,(R5)+                ; MOVE LO DBN TO CHAIN
006433  100251                                  MOV     R1,(R5)+                ; MOVE HI DBN TO CHAIN
006434  115407                                  INC     R0                      ; INCREMENT LO DBN
006435  046437                                  BCC     FCARY                   ; IF NO CARRY, BRANCH
006436  115401                                  INC     R1                      ; INCREMENT HI DBN
006437  117404                          FCARY:  DEC     R4                      ; DECREMENT SECTOR COUNT
006440  056431                                  BNE     FLOOP                   ; BRANCH IF COUNT UNEXHAUSTED
006441  104207  100000                          MOV     #FSTOP,R0               ; GET FORMAT END-OF-CHAIN FLAG
006443  100157                                  MOV     R0,(R5)                 ; MOVE INTO CHAIN
006444  104302  002636                          MOV     SDI,R2                  ; MOVE MASK TO R2
006446  020754                                  CALL    RDSTAT                  ; GET DRIVE STATUS
006447  115002                                  TST     R2                      ; WAS IT OK?
006450  016516                                  BEQ     2$                      ; IF NO ERROR, BRANCH
006451  102201  000400                          BIT     #RCVERR,R1              ; SEE IF ANY ERRORS
006453  056475                                  BNE     1$                      ; IF NOT, ERROR
006454                                          HARDER  10                      ; REPORT INVALID STATUS ERROR
006454  104200  000341  001037                                  MOV     #MS10,OUT.04
006457  104300  001475  001036                                  MOV     LUNIT,OUT.03
006462  104202  105702                                          MOV     #10!ERHARD+3000.,R2
006464  104020  001035                                          MOV     R2,OUT.02
006466  104200  006466  001034                                  MOV     #.,OUT.01
006471  104200  060013  001033                                  MOV     #ERRMES,OUT.RQ
006474  001732                                  BR      TESTEW          ; BRANCH TO DONE
006475                                  1$:
006475                                          HARDER  12                      ; REPORT XMIT ERROR
006475  104200  000434  001037                                  MOV     #MS12,OUT.04
006500  104300  001475  001036                                  MOV     LUNIT,OUT.03
006503  104202  105704                                          MOV     #12!ERHARD+3000.,R2
006505  104020  001035                                          MOV     R2,OUT.02
006507  104200  006507  001034                                  MOV     #.,OUT.01
006512  104200  060013  001033                                  MOV     #ERRMES,OUT.RQ
006515  001732                                  BR      TESTEW                  ; BRANCH TO DONE
006516                                  2$:
006516  102201  100000                          BIT     #RWRDY,R1               ; TEST R/W READY
006520  056542                                  BNE     FGO                     ; IF ASSERTED, BRANCH
006521                                          HARDER  18                      ;READ/WRITE READY DROPPED BEFORE FORMAT
006521  104200  000737  001037                                  MOV     #MS18,OUT.04
006524  104300  001475  001036                                  MOV     LUNIT,OUT.03
```

```
        006527  104202  105712                                                  MOV     #18!ERHARD+3000.,R2
        006531  104020  001035                                                  MOV     R2,OUT.02
        006533  104200  006533  001034                                          MOV     #.,OUT.01
        006536  104200  060013  001033                                          MOV     #ERRMES,OUT.RQ
    44  006541  001740                          BR      TESTEV                  ; BRANCH TO EXIT
    45  006542  104207  003652          FGO:    MOV     #FCHAIN,R0              ; POINT TO FORMAT CHAIN (FOR XFC)
    46  006544  104304  003000                  MOV     SUB+DATPRE,R4           ; GET DATA PREAMBLE LENGTHS
    47  006546  103204  177400                  BIC     #HIBYTE,R4              ; CLEAR UNUSED BITS
    48  006550  104303  003000                  MOV     SUB+HDRPRE,R3           ; GET HEADER PREAMBLE LENGTH
    49  006552  110703                          SWAB    R3                      ; SWAP BYTES
    50  006553  103203  177400                  BIC     #HIBYTE,R3              ; CLEAR UNUSED BYTES
    51  006555  104301  003076                  MOV     CURTRK,R1               ; TRACK NUMBER
    52  006557  104302  002636                  MOV     SDI,R2                  ; SET PORT INDICATOR
    53  006561  060001                          XFC     FORMAT                  ; FORMAT THE TRACK (BUFFER CONTENTS ARE A DON'T CARE
    54  006562  115001                          TST     R1                      ; SEE IF ANY ERRORS OCCURRED
    55  006563  016623                          BEQ     FOREXT                  ; IF NOT, BRANCH
    56  006564                                  HARDER  19,<INS+1,INS+2,INS+3,CURTRK> ;TIMEOUT OF DRIVE OR READ/WRITE READY DROPPED
        006564  104200  000771  001037                                          MOV     #MS19,OUT.04
        006567  104300  002741  001040                                          MOV     INS+1,OUT.05
        006572  104300  002742  001041                                          MOV     INS+2,OUT.06
        006575  104300  002743  001042                                          MOV     INS+3,OUT.07
        006600  104300  003076  001043                                          MOV     CURTRK,OUT.08
        006603  104300  001475  001036                                          MOV     LUNIT,OUT.03
        006606  104202  105713                                                  MOV     #19!ERHARD+3000.,R2
        006610  104020  001035                                                  MOV     R2,OUT.02
        006612  104200  006612  001034                                          MOV     #.,OUT.01
        006615  104200  060013  001033                                          MOV     #ERRMES,OUT.RQ
    57  006620  104307  001033                  MOV     OUT.RQ,R0               ; PRINT ERROR
    58  006622  021000                          CALL    HOSTRQ
    59  006623  000000          FOREXT: RETURN
```

```
                                                .SBTTL  TRKTST - TEST TRACK
006624                                  TRKTST:
                                        ;
                                        ;       TEST THE ENTIRE TRACK UNTIL AT LEAST ONE BLOCK IS SUCCESSFULLY
                                        ;       WRITTEN AND READ WITH SEVERAL DATA PATTERNS
                                        ;
006624  104307  003061                          MOV     TSTBLK,R0               ; MOVE LO STARTING DBN TO CURRENT DBN
006626  104070  003071                          MOV     R0,CURBLK
006630  104307  003062                          MOV     TSTBLK+1,R0             ; MOVE HI STARTING DBN TO CURRENT DBN
006632  104070  003072                          MOV     R0,CURBLK+1
006634  104301  003060                          MOV     SECTRK,R1               ; GET SECTORS/TRACK
006636  117401                                  DEC     R1                      ; ADJUST FOR END LOOP WHEN NEGATIVE
006637  104010  003100                  TRKLOP: MOV     R1,SECCNT               ; SAVE SECTORS LEFT IN SECTOR COUNT
006641  104201  000001                          MOV     #1,R1                   ; MOVE 1 TO CURRENT PATTERN
006643  026666                                  CALL    WTRCMP                  ; WRITE THAN READ AND COMPARE THE SECTOR
006644  115007                                  TST     R0                      ; SEE IF WRITES, READS AND COMPARES WERE GOOD
006645  016665                                  BEQ     TRKEXT                  ; IF SO, BRANCH
006646  104307  003071                          MOV     CURBLK,R0               ; GET LO CURRENT BLOCK NUMBER
006650  115407                                  INC     R0                      ; INCREMENT DBN NUMBER
006651  104070  003071                          MOV     R0,CURBLK               ; SAVE
006653  046661                                  BCC     TRKBOT                  ; BRANCH IF NO CARRY
006654  104307  003072                          MOV     CURBLK+1,R0             ; GET HI DBN NUMBER
006656  115407                                  INC     R0                      ; INCREMENT
006657  104070  003072                          MOV     R0,CURBLK+1             ; SAVE
006661  104301  003100                  TRKBOT: MOV     SECCNT,R1               ; GET SECTOR COUNT
006663  117401                                  DEC     R1                      ; DECREMENT COUNT
006664  036637                                  BPL     TRKLOP                  ; BRANCH IF COUNT POSITIVE
006665  000000                          TRKEXT: RETURN
```

```
                                                .SBTTL  WRTCMP - WRITE A DATA PATTERN AND COMPARE
006666                                  WTRCMP:
                                        ;
                                        ;       WRITE A DATA PATTERN TO A SECTOR, READ IT BACK, THEN DO A DATA
                                        ;       COMPARE ON THE DATA.  DO THIS AS MANY TIMES AS THERE IS PATTERNS.
                                        ;       CURRENT PATTERN NUMBER IN 'CURPAT'
                                        ;
006666  104010  003077                          MOV     R1,CURPAT               ; R1 IS CURRENT PATTERN NUMBER
006670  105201  003132                          ADD     #PATPTR,R1              ; R1 NOW POINTS TO PATTERN TO GENERATE
006672  026713                                  CALL    GENPAT                  ; GENERATE THE PATTERN IN THE OUTPUT BUFFER
006673  026734                                  CALL    WRITEB                  ; WRITE THE PATTERN
006674  115007                                  TST     R0                      ; SEE IF ANY ERROR OCCURRED
006675  056712                                  BNE     WTREXT                  ; IF SO, BRANCH
006676  027175                                  CALL    READB                   ; READ THE BLOCK BACK
006677  115007                                  TST     R0                      ; SEE IF ANY ERRORS OCCURRED
006700  056712                                  BNE     WTREXT                  ; IF SO, BRANCH
006701  027402                                  CALL    CMPDAT                  ; COMPARE THE DATA
006702  115007                                  TST     R0                      ; SEE IF ANY ERRORS OCCURRED
006703  056712                                  BNE     WTREXT                  ; IF SO, BRANCH
006704  104301  003077                          MOV     CURPAT,R1               ; GET CURRENT PATTERN
006706  115401                                  INC     R1                      ; NEXT PATTERN
006707  106301  003132                          CMP     PATPTR,R1               ; COMARE AGAINST MAXIMUM PATTERN NUMBER
006711  036666                                  BPL     WTRCMP                  ; IF LESS OR EQUAL, LOOP
006712  000000                          WTREXT: RETURN
```

```
      1
      2 006713                                GENPAT:
      3
      4                                       ;
      5                                       ;       GENERATE THE DATA PATTERN IN THE OUTPUT BUFFER
      6                                       ;
      7                                       ;       R1 POINTS TO POINTER TO PATTERN BUFFER WHICH IS PATTERN LENGTH
      8                                       ;       (1 WORD) FOLLOWED BY THAT MANY WORDS OF PATTERN
      9 006713                                ;
        006713  100462                                PUSH    R2                      ; SAVE R2
                                                                                              MOV R2,-(SP)
     10 006714  104117                                MOV     (R1),R0                 ; R0 POINTS TO START OF PATTERN BUFFER
     11 006715  104203  000400                        MOV     #256.,R3                ; R3 HOLDS PATTERN COUNT
     12 006717  104204  003243                        MOV     #OBUFF,R4               ; R4 POINTS TO OUTPUT BUFFER
     13 006721  104071                        GENOUT: MOV     R0,R1                   ; R1 POINTS TO START OF PATTERN BUFFER
     14 006722  104215                                MOV     (R1)+,R5                ; R5 CONTAINS LENGTH OF PATTERN
     15 006723  104212                        GENIN:  MOV     (R1)+,R2                ; GET WORD OF PATTERN
     16 006724  100242                                MOV     R2,(R4)+                ; MOVE TO BUFFER
     17 006725  117403                                DEC     R3                      ; DECREMENT BUFFER COUNT
     18 006726  016732                                BEQ     GENEXT                  ; IF BUFFER FULL, EXIT
     19 006727  117405                                DEC     R5                      ; DECREMENT PATTERN COUNT
     20 006730  056723                                BNE     GENIN                   ; IF NON-ZERO, BRANCH
     21 006731  006721                                BR      GENOUT                  ; START PATTERN OVER AGAIN
     22 006732                                GENEXT: POP     R2                      ; RESTORE R2
        006732  104262                                                                       MOV (SP)+,R2
     23 006733  000000                                RETURN
```

```
                                                .SBTTL  WRITEB - WRITE A SECTOR
006734                                  WRITEB:
                                        ;
                                        ;       WILL WRITE ONE SECTOR OF DATA TO THE DISK
                                        ;
006734  104207  140000                          MOV     #WSTOP,R0
006736  104070  003124                          MOV     R0,RW.STAT+CHAIN; MOVE END-OF-CHAIN TO CHAIN
006740  104307  003071                          MOV     CURBLK,R0               ; MOVE LO DBN TO CHAIN
006742  104070  003126                          MOV     R0,RW.LOW+CHAIN
006744  104307  003072                          MOV     CURBLK+1,R0             ; MOVE HI DBN TO CHAIN
006746  104301  002776                          MOV     SUB+HIDBN,R1            ; GET HIGH ORDER BITS OF STARTING DBN
006750  110601                                  ROR     R1                      ; MOVE TO CORRECT POSITION
006751  110601                                  ROR     R1
006752  110601                                  ROR     R1
006753  110601                                  ROR     R1
006754  103201  170377                          BIC     #HBLONB,R1              ; CLEAR UNUSED BITS
006756  101201  140000                          BIS     #HD.DBN,R1              ; SET HEADER CODE
006760  101017                                  BIS     R1,R0                   ; SET IN R0
006761  104070  003127                          MOV     R0,RW.HI+CHAIN
006763  104307  003076                          MOV     CURTRK,R0               ; MOVE TRACK + RTC TO CHAIN
006765  101207  122400                          BIS     #WREAL,R0               ; SET REAL TIME COMMAND WRITTE
006767  104070  003130                          MOV     R0,RW.CMD+CHAIN
006771  104207  003243                          MOV     #OBUFF,R0               ; MOVE OUTPUT BUFFER TO CHAIN
006773  104070  003125                          MOV     R0,RW.BUF+CHAIN
006775  104302  002636                          MOV     SDI,R2                  ; MOVE MASK TO R2
006777  020754                                  CALL    RDSTAT                  ; GET DRIVE STATUS
007000  115002                                  TST     R2                      ; WAS IT OK?
007001  017047                                  BEQ     2$                      ; IF NO ERROR, BRANCH
007002  102201  000400                          BIT     #RCVERR,R1              ; SEE IF ANY ERRORS
007004  057026                                  BNE     1$                      ; IF NOT, ERROR
007005                                          HARDER  10                      ; REPORT INVALID STATUS ERROR
007005  104200  000341  001037                          MOV     #MS10,OUT.04
007010  104300  001475  001036                          MOV     LUNIT,OUT.03
007013  104202  105702                                  MOV     #10!ERHARD+3000.,R2
007015  104020  001035                                  MOV     R2,OUT.02
007017  104200  007017  001034                          MOV     #.,OUT.01
007022  104200  060013  001033                          MOV     #ERRMES,OUT.RQ
007025  001732                                  BR      TESTEW                  ; BRANCH TO DONE
007026                                  1$:
007026                                          HARDER  12                      ; REPORT XMIT ERROR
007026  104200  000434  001037                          MOV     #MS12,OUT.04
007031  104300  001475  001036                          MOV     LUNIT,OUT.03
007034  104202  105704                                  MOV     #12!ERHARD+3000.,R2
007036  104020  001035                                  MOV     R2,OUT.02
007040  104200  007040  001034                          MOV     #.,OUT.01
007043  104200  060013  001033                          MOV     #ERRMES,OUT.RQ
007046  001732                                  BR      TESTEW                  ; BRANCH TO DONE
007047                                  2$:
007047  102201  100000                          BIT     #RWRDY,R1               ; SEE IF READ/WRITE READY IS ASSERTED
007051  057100                                  BNE     WGO                     ; IF SO, BRANCH
007052  104307  003100                          MOV     SECCNT,R0               ; GET SECTOR COUNT
007054  057174                                  BNE     WRTEXT                  ; IF NONZERO, DO NOT REPORT ERROR
007055                                          HARDER  22                      ;READ/WRITE DROPPED READY BEFORE WRITE
007055  104200  001117  001037                          MOV     #MS22,OUT.04
007060  104300  001475  001036                          MOV     LUNIT,OUT.03
007063  104202  105716                                  MOV     #22!ERHARD+3000.,R2
007065  104020  001035                                  MOV     R2,OUT.02
```

```
                007067  104200  007067  001034                                          MOV     #.,OUT.01
                007072  104200  060013  001033                                          MOV     #ERRMES,OUT.RQ
     42 007075  104207  000001                      MOV     #1,R0                   ; FLAG ERROR
     43 007077  007174                              BR      WRTEXT                  ; BRANCH
     44 007100  104302  002636          WGO:        MOV     SDI,R2                  ; SET PORT INDICATOR
     45 007102  060012                              XFC     WAITSI                  ; WAIT FOR SECTOR OR INDEX PULSE
     46 007103  104207  003124                      MOV     #CHAIN,R0               ; POINT TO WRITE CHAIN
     47 007105  104304  003000                      MOV     SUB+DATPRE,R4           ; MOVE DATA PREAMBLE LENGTH TO R4
     48 007107  103204  177400                      BIC     #HIBYTE,R4              ; CLEAR UNUSED BITS
     49 007111  060003                              XFC     XWRITE                  ; WRITE THE BLOCK
     50 007112  114007                              CLR     R0                      ; FLAG AS NO ERRORS
     51 007113  115000  003104                      TST     WFLAG                   ; DID WE TEST WHEN WRITE PROTECTED?
     52 007115  057174                              BNE     WRTEXT                  ; IF SO, EXIT
     53 007116  115001                              TST     R1                      ; SEE IF AN ERROR OCCURRED
     54 007117  017174                              BEQ     WRTEXT                  ; IF NOT, BRANCH
     55 007120  104307  003100                      MOV     SECCNT,R0               ; SEE IF ERROR SHOULD BE REPORTED
     56 007122  057174                              BNE     WRTEXT                  ; IF NOT, BRANCH
     57 007123                                      HARDER  23,<R1,CURBLK,CURBLK+1,INS+1,INS+2,INS+3,CURTRK> ;ERROR DURING WRITE
                007123  104200  001151  001037                                          MOV     #MS23,OUT.04
                007126  104010  001040                                                  MOV     R1,OUT.05
                007130  104300  003071  001041                                          MOV     CURBLK,OUT.06
                007133  104300  003072  001042                                          MOV     CURBLK+1,OUT.07
                007136  104300  002741  001043                                          MOV     INS+1,OUT.08
                007141  104300  002742  001044                                          MOV     INS+2,OUT.09
                007144  104300  002743  001045                                          MOV     INS+3,OUT.10
                007147  104300  003076  001046                                          MOV     CURTRK,OUT.11
                007152  104300  001475  001036                                          MOV     LUNIT,OUT.03
                007155  104202  105717                                                  MOV     #23!ERHARD+3000.,R2
                007157  104020  001035                                                  MOV     R2,OUT.02
                007161  104200  007161  001034                                          MOV     #.,OUT.01
                007164  104200  060013  001033                                          MOV     #ERRMES,OUT.RQ
     58 007167  104307  001033                      MOV     OUT.RQ,R0
     59 007171  021000                              CALL    HOSTRQ
     60 007172  104207  000001                      MOV     #1,R0                   ; MAKE R0 NONZERO TO REPORT ERROR
     61 007174  000000          WRTEXT: RETURN
```

```
                                                .SBTTL  READB - READ ONE SECTOR
007175                                  READB:
                                        ;
                                        ;       READ ONE SECTOR FROM DISK
                                        ;
007175  104207  100000                          MOV     #RSTOP,R0               ; MOVE END-OF-CHAIN TO CHAIN
007177  104070  003124                          MOV     R0,RW.STAT+CHAIN
007201  104307  003076                          MOV     CURTRK,R0               ; MOVE TRACK NUMBER AND RTC TO CHAIN
007203  101207  013400                          BIS     #RREAL,R0               ; SET REAL TIME READ COMMMAND
007205  104070  003130                          MOV     R0,RW.CMD+CHAIN
007207  104207  003652                          MOV     #RBUFD,R0               ; MOVE POINTER TO INPUT BUFFER INTO CHAIN
007211  104070  003125                          MOV     R0,RW.BUF+CHAIN
007213  104302  002636                          MOV     SDI,R2                  ; MOVE MASK TO R2
007215  020754                                  CALL    RDSTAT                  ; GET DRIVE STATUS
007216  115002                                  TST     R2                      ; WAS IT OK?
007217  017265                                  BEQ     2$                      ; IF NO ERROR, BRANCH
007220  102201  000400                          BIT     #RCVERR,R1              ; SEE IF ANY ERRORS
007222  057244                                  BNE     1$                      ; IF NOT, ERROR
007223                                          HARDER  10                      ; REPORT INVALID STATUS ERROR
007223  104200  000341  001037                                  MOV     #MS10,OUT.04
007226  104300  001475  001036                                  MOV     LUNIT,OUT.03
007231  104202  105702                                          MOV     #10!ERHARD+3000.,R2
007233  104020  001035                                          MOV     R2,OUT.02
007235  104200  007235  001034                                  MOV     #.,OUT.01
007240  104200  060013  001033                                  MOV     #ERRMES,OUT.RQ
007243  001732                                  BR      TESTEW                  ; BRANCH TO DONE
007244                                  1$:
007244                                          HARDER  12                      ; REPORT XMIT ERROR
007244  104200  000434  001037                                  MOV     #MS12,OUT.04
007247  104300  001475  001036                                  MOV     LUNIT,OUT.03
007252  104202  105704                                          MOV     #12!ERHARD+3000.,R2
007254  104020  001035                                          MOV     R2,OUT.02
007256  104200  007256  001034                                  MOV     #.,OUT.01
007261  104200  060013  001033                                  MOV     #ERRMES,OUT.RQ
007264  001732                                  BR      TESTEW                  ; BRANCH TO DONE
007265                                  2$:
007265  102201  100000                          BIT     #RWRDY,R1               ; SEE IF READ/WRITE READY IS ASSERTED
007267  057314                                  BNE     RGO                     ; IF SO, BRANCH
007270  104307  003100                          MOV     SECCNT,R0               ; GET SECTOR COUNT
007272  057401                                  BNE     REDEXT                  ; IF NONZERO, DO NOT REPORT ERROR
007273                                          HARDER  24                      ;READ/WRITE DROPPED READY BEFORE READ
007273  104200  001306  001037                                  MOV     #MS24,OUT.04
007276  104300  001475  001036                                  MOV     LUNIT,OUT.03
007301  104202  105720                                          MOV     #24!ERHARD+3000.,R2
007303  104020  001035                                          MOV     R2,OUT.02
007305  104200  007305  001034                                  MOV     #.,OUT.01
007310  104200  060013  001033                                  MOV     #ERRMES,OUT.RQ
007313  007374                                  BR      REDERR                  ; GO PRINT THE ERROR
007314  104302  002636                  RGO:    MOV     SDI,R2                  ; SET PORT INDICATOR
007316  060012                                  XFC     WAITSI                  ; WAIT FOR SECTOR OR INDEX PULSE
007317  104207  003124                          MOV     #CHAIN,R0               ; POINT TO READ CHAIN
007321  060002                                  XFC     XREAD                   ; READ THE SECTOR
007322  114007                                  CLR     R0                      ; FLAG AS NO ERRORS
007323  115001                                  TST     R1                      ; SEE IF ERROR OCCURRED
007324  017401                                  BEQ     REDEXT                  ; IF NOT, BRANCH
007325  104307  003100                          MOV     SECCNT,R0               ; SEE IF ERROR SHOULD BE REPORTED
007327  057401                                  BNE     REDEXT                  ; IF NOT, BRANCH
```

```
     40 007330                                          HARDER  25,<R1,CURBLK,CURBLK+1,INS+1,INS+2,INS+3,CURTRK> ;ERROR DURING READ
        007330  104200  001337  001037                                                    MOV     #MS25,OUT.04
        007333  104010  001040                                                            MOV     R1,OUT.05
        007335  104300  003071  001041                                                    MOV     CURBLK,OUT.06
        007340  104300  003072  001042                                                    MOV     CURBLK+1,OUT.07
        007343  104300  002741  001043                                                    MOV     INS+1,OUT.08
        007346  104300  002742  001044                                                    MOV     INS+2,OUT.09
        007351  104300  002743  001045                                                    MOV     INS+3,OUT.10
        007354  104300  003076  001046                                                    MOV     CURTRK,OUT.11
        007357  104300  001475  001036                                                    MOV     LUNIT,OUT.03
        007362  104202  105721                                                            MOV     #25!ERHARD+3000.,R2
        007364  104020  001035                                                            MOV     R2,OUT.02
        007366  104200  007366  001034                                                    MOV     #.,OUT.01
        007371  104200  060013  001033                                                    MOV     #ERRMES,OUT.RQ
     41 007374  104307  001033          REDERR: MOV     OUT.RQ,R0
     42 007376  021000                          CALL    HOSTRQ
     43 007377  104207  000001                  MOV     #1,R0                   ; FLAG ERROR
     44 007401  000000          REDEXT: RETURN
```

```
                                                .SBTTL  CMPDAT - COMPARE DATA
007402                                  CMPDAT:
                                        ;
                                        ;       COMPARE THE DATA IN 'OBUFF' (OUTPUT BUFFER) WITH 'RBUFD'
                                        ;       (INPUT BUFFER)
                                        ;
007402  104207  003243                          MOV     #OBUFF,R0               ; R0 POINTS AT OUTPUT BUFFER
007404  104201  003652                          MOV     #RBUFD,R1               ; R1 POINTS AT INPUT BUFFER
007406  114003                                  CLR     R3                      ; COUNTER
007407  104274                          CMPLOP: MOV     (R0)+,R4                ; GET OUTPUT BUFFER WORD
007410  106214                                  CMP     (R1)+,R4                ; COMPARE AGAINST INPUT BUFFER
007411  017465                                  BEQ     NOERR                   ; IF NO ERROR, BRANCH
007412  104305  003100                          MOV     SECCNT,R5               ; SEE IF ERROR IS TO BE REPORTED
007414  057472                                  BNE     CMPEXT                  ; IF NOT, BRANCH
007415                                          HARDER  26.,<R3,-(R0),-(R1),INS+1,INS+2,INS+3,CURTRK> ;DATA COMPARE FAILURE
007415  104200  001473  001037                                                          MOV     #MS26,OUT.04
007420  104030  001040                                                                  MOV     R3,OUT.05
007422  104470  001041                                                                  MOV     -(R0),OUT.06
007424  104410  001042                                                                  MOV     -(R1),OUT.07
007426  104300  002741  001043                                                          MOV     INS+1,OUT.08
007431  104300  002742  001044                                                          MOV     INS+2,OUT.09
007434  104300  002743  001045                                                          MOV     INS+3,OUT.10
007437  104300  003076  001046                                                          MOV     CURTRK,OUT.11
007442  104300  001475  001036                                                          MOV     LUNIT,OUT.03
007445  104202  105722                                                                  MOV     #26!ERHARD+3000.,R2
007447  104020  001035                                                                  MOV     R2,OUT.02
007451  104200  007451  001034                                                          MOV     #.,OUT.01
007454  104200  060013  001033                                                          MOV     #ERRMES,OUT.RQ
007457  104307  001033                                  MOV     OUT.RQ,R0
007461  021000                                          CALL    HOSTRQ
007462  104207  000001                                  MOV     #1,R0                   ; SET FOR ERROR
007464  007472                                          BR      CMPEXT                  ; EXIT
007465  115403                          NOERR:          INC     R3                      ; INCREMENT COUNT
007466  106203  000400                                  CMP     #256.,R3                ; SEE IF COMPARE COMPLETE
007470  057407                                          BNE     CMPLOP                  ; IF NOT, BRANCH
007471  114007                                          CLR     R0                      ; FLAG AS NO ERRORS
007472  000000                          CMPEXT: RETURN

        002224                          OVL.TS  =       .-T7

007473                                          $MEND
007473  055136                                  .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
        000001                                  .END
```

SYMBOL TABLE

```
AREA    003101          ECC    = 000015          HBHINB= 007777          MS.GST  002443          OUT.03  001036
ATTN  = 000002          ECCFLG= 010000          HBLONB= 170377          MS.INR  002466          OUT.04  001037
AVAIL = 000100          ECCRSH= 000002          HDRPRE= 000005          MS.MOD  002452          OUT.05  001040
BF.DAT= 000000          ECHO   = 000010          HD.BAD= 110000          MS.ONL  002360          OUT.06  001041
BF.ECC= 000401          ECHOC  = 000350          HD.DBN= 140000          MS.RUN  002502          OUT.07  001042
BF.EDC= 000400          ENDGTU  004273          HD.LBN= 000000          MS.SCR  002423          OUT.08  001043
BLOCKC  003070          ENDPNT= 005247          HD.PRV= 050000          MS.SEK  002462          OUT.09  001044
BLOCKG  003067          EOC    = 100000          HD.RBN= 060000          MS.STR= 070400          OUT.10  001045
BLOCKT  003066          ERECOV= 000006          HD.REV= 030000          MS1     000000          OUT.11  001046
BREAK = 000000          ERHARD= 100000          HD.XBN= 120000          MS10    000341          OUT.12  001047
BUFFLG= 040000          ERLEV  = 000002          HEADER= 000002          MS11    000366          OUT.13  001050
BUFSIZ= 000043          ERRMC  = 060014          HIBYTE= 177400          MS12    000434          OUT.14  001051
CHAIN   003124          ERRMES= 060013          HICYL  = 000001          MS13    000461          OUT.15  001052
CHECK = 000010          ERRN   = 005670          HIDBN  = 000003          MS14    000501          OUT.16  001053
CHGMOD= 000201          ERRORS  002731          HILBN  = 000002          MS15    000555          OUT.17  001054
CHRRES= 000170          ERRPOS  003103          HIMEM  = 007774          MS16    000635          OUT.18  001055
CMPDAT  007402          ERSOFT= 140000          HIRBN  = 000003          MS17    000677          OUT.19  001056
CMPEXT  007472          EXIT   = 000021          HIXBN  = 000002          MS18    000737          OUT.20  001057
CMPLOP  007407          FB.DAT= 000000          HOSTRQ  001000          MS19    000771          OUT.21  001060
COMND   002637          FB.EDC= 000400          HRDRFV= 000003          MS2     000014          OUT.22  001061
COMPAR= 000006          FCARY   006437          INDEXS  003123          MS20    001050          OUT.23  001062
COMPLT= 000176          FCHAIN  003652          INR     002737          MS2000  002510          OUT.24  001063
CR      002760          FCTSIZ= 000010          INS     002740          MS21    001073          OUT.25  001064
CR.CLR  002645          FDIACY  003054          INSEEK= 000012          MS22    001117          OUT.26  001065
CR.DIS  002652          FGO     006542          INTEDC= 000105          MS23    001151          OUT.27  001066
CR.GCR  002657          FLAG    003105          IRECLB= 000216          MS24    001306          OUT.28  001067
CR.GST  002671          FLOOP   006431          LARGE  = 000001          MS25    001337          OUT.29  001070
CR.INR  002710          FNDCYL  001770          LBHINB= 177417          MS26    001473          OUT.30  001071
CR.MOD  002676          FNDCY2  001776          LBLONB= 177760          MS27    001633          OUT.31  001072
CR.ONL  002640          FND3    002006          LBNCYL= 000000          MS28    001716          OUT.32  001073
CR.RUN  002715          FND4    002034          LBNHST= 000012          MS29    002007          OUT.33  001074
CR.SCR  002664          FOREXT  006623          LBNTRK= 000011          MS3     000032          OUT.34  001075
CR.SEK  002703          FORMAT= 000001          LCDBN   003047          MS30    002053          OVERFL= 000002
CURBLK  003071          FORTRK  006405          LDIACY  003051          MS31    002077          OVE.MN= 000714
CURGRP  003075          FOR.SZ= 001375          LETTER  003102          MS32    002130          OVE.MS= 000000
CURPAT  003077          FRAME  = 000004          LINIT   001476          MS33    002156          OVE.SU= 005247
CURTRK  003076          FSTOP  = 100000          LINKLN= 000007          MS34    002220          OVE.TS= 005247
CVT    = 000020          FTLDEV= 040000          LOBYTE= 000377          MS35    002245          OVL.MN= 004334
CYLLO   003116          FTLSYS= 000000          LONG   = 002707          MS4     000063          OVL.MS= 003447
D2HARD= 040000          FT.BUF= 000000          LONGTO= 000001          MS5     000110          OVL.SU= 000766
DATPRE= 000005          FT.HI  = 000001          LOW    = 000002          MS6     000135          OVL.TS= 002225
DBNCYL= 000022          FT.LOW= 000002          LUNIT   001475          MS7     000160          OVL...= 013216
DCLOCK= 000004          FXBNCY  003056          MAXSND= 001750          MS8     000221          OVRLAY  001633
DIA     002724          GCR     002732          MEDTYP= 000006          MS9     000306          OVRLNM  001730
DINIT = 000011          GENEXT  006732          MESSAG= 060015          MWR    = 000017          OVRLPT= 005247
DIS     002726          GENIN   006723          MICREV= 000003          NOERR   007465          OVRPNT  001731
DISCON= 000204          GENOUT  006721          MOD     002744          NSCSL   003020          OVSTRT= 007774
DMSDI   003017          GENPAT  006713          MODE    002745          NUMOVL= 000003          OVS.MN= 001040
DONE   = 060016          GETCHR= 000207          MODE1   002746          NUMPTR= 000014          OVS.MS= 011730
DONECD  001627          GETSTA= 000011          MODE2   002747          OBUFF   003243          OVS.SU= 021046
DRC     002730          GETSUB= 000210          MRD    = 000016          OCNT$  = 000003          OVS.TS= 023022
DRTYPE= 000007          GETU    003652          MSSG$  = 000000          OLDGRP  003107          OV...  = 013636
DRVCLR= 000005          GROUP   003120          MS.CLR  002365          ONL     002722          PATPTR  003132
DRVID  = 000004          GRPCNT  003106          MS.CNT= 152000          OTABLE  001547          PAT4    003137
DRVONL= 000213          GRPCYL= 000002          MS.DIS  002375          OUT.RQ  001033          PAT5    003160
DRVRUN= 000014          GRPOFF= 000011          MS.END= 131000          OUT.01  001034          PAT6    003201
DR.CLR  001761          GST     002735          MS.GCR  002404          OUT.02  001035          PAT8    003222
```

UDAT3 DISK FUNCTIONAL DMACR X04.01 13-APR-82 18:08:21 PAGE 63-2
SYMBOL TABLE

```
PHYSA = 001000          SCR     002733          SNDLV1  006250          TO      001463          T6F     006233
PORT2   001556          SCTWRD= 000377          SNDL1A  006253          TOOBIG= 000001          T6G     006173
PORT3   001606          SDI     002636          SNDONE= 002644          TRACK   003121          T7      005247
PORT4   001614          SDILTO  001462          SS    = 000001          TRKBOT  006661          T7A     005262
PORT5   001623          SDISTO  001461          SSCTOR  003122          TRKEXT  006665          T7C     005321
RBNTRK= 000004          SDIVER= 000000          ST      002751          TRKGRP= 000003          T7X     005470
RBUFD   003652          SECCNT  003100          STACK   001542          TRKLOP  006637          T8      005545
RBUFLN= 000415          SECCYL  003074          START   003652          TRKTST  006624          T8A     005610
RBUFO   003644          SECGRP  003073          STATUS= 000007          TSTBLK  003061          T8B     005626
RCONT = 000000          SECTRK  003060          STRST   002131          TSTCMD  006263          T8D     005660
RCTCPS= 000001          SEEK    002156          STRTST  005247          TSTCME  006403          T8E     005707
RCTCSZ= 000014          SEEKA   002171          STSRES= 000366          TSTCM2  006265          T8F     005770
RCV   = 000005          SEEK1   002203          ST.C  = 000002          TSTCYL  003063          T8T     005700
RCVERR= 000400          SEEK2   002204          ST.CON= 000002          T0A     005373          T9      006123
RCVRDY= 000001          SEEK3   002321          ST.DB = 001000          T0B     005400          UDADM3= 001000 G
RDSTAT  000754          SEND  = 000004          ST.DF = 000020          T0B     005570          UNITNB  002635
READB   007175          SER00   002347          ST.DR = 000040          T1MSIZ= 060000          UNITS   002615
READ1   002325          SER10   002626          ST.EL = 000010          T10     006134          UNIT0 = 000001
READ1A  002412          SER11   002641          ST.ERR= 000002          T10LOP  006135          UNIT1 = 000002
READ1B  002536          SER12   002657          ST.FE = 000200          T11     005771          UNIT2 = 000004
READ1C  002515          SER13   002671          ST.FO = 002000          T11ER   006277          UNIT3 = 000010
READ1E  002537          SER14   002704          ST.MOD= 000001          T11X1   006333          UNSSUC= 000175
READ1X  002607          SER15   002732          ST.MSK= 000000          T11X2   006357          UREAD = 000013
REDERR  007374          SER16   002754          ST.OA = 000200          T12     006241          UTOTST= 060012
REDEXT  007401          SER17   002777          ST.PE = 000040          T2CMD = 060002          UWRITE= 000014
RETS  = 000001          SER18   003031          ST.PS = 000002          T2DLL = 060001          U.SUBU= 000050
REVECT= 000004          SER18A  003050          ST.RE = 000100          T3A     005532          U.UNUM= 000063
REVS  = 000007          SER18B  003045          ST.RR = 000100          T3B     005561          VAR1    003110
RGO     007314          SER18C  003042          ST.RTY= 000003          T3C     005456          VAR2    003112
RM    = 000004          SER18D  003037          ST.RU = 000001          T3D     005562          VAR3    003114
ROFDBN  003042          SER18E  001543          ST.SR = 000020          T3STRT  001555          VAR4    003115
ROFDC   003044          SER22   003052          ST.STA= 000001          T4A     005610          WAITSi= 000012
RREAL = 013400          SER23   003120          ST.S7 = 000400          T4A1    005653          WBUFLN= 000401
RSTOP = 100000          SER39   002306          ST.UNT= 000000          T4B     005707          WCONT = 040000
RTDS    000720          SER40   003134          ST.WE = 000010          T4BB1 = 060005          WFLAG   003104
RTDSL   000746          SER41   003156          SUB   = 002773          T4BB2 = 060006          WGO     007100
RUN     002736          SER50   003171          SUBUNT  002734          T4C     005742          WREAL = 122400
RUNCMD  002750          SER51   003176          T       005317          T4D     005746          WRITEB  006734
RUNLBC= 000014          SER52   003204          TALK    001076          T4MPRM= 060003          WRONG = 000002
RWRDY = 100000          SER53   003243          TEST  = 000002          T4MXFR= 060011          WRTEXT  007174
RW.ANG= 000006          SER54   003274          TESTED  001753          T4SEEK= 060010          WSTOP = 140000
RW.BUF= 000001          SER55   003325          TESTEV  001740          T4SOFT= 060007          WTRCMP  006666
RW.CMD= 000004          SER56   003360          TESTEW  001732          T4UPRM= 060004          WTREXT  006712
RW.HI = 000003          SER57   003410          TESTEX  001746          T6      005761          XBNCYL= 000021
RW.LOW= 000002          SETUP = 000001          TESTEY  001752          T6A     005770          XFERRT= 000000
RW.SDI= 000005          SHRTTO= 000000          TESTX   001617          T6C     006030          XREAD = 000002
RW.STA= 000000          SL.GRP= 107000          TEST4 = 000000          T6D     006034          XWRITE= 000003
SBCRES= 000167          SNDAGN  001010          TIMEOU= 000001          T6E     006217

. ABS.  027474      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  5145 WORDS  ( 21 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  70 PAGES
B:UDAT3.OBK,B:UDAT3/C=$DMACRO,B:UDAT3
```

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

```
        61-34    61-41    61-57    62-19    62-22    62-29    62-40    63-15
MAXSND   4-48#    25-13    37-73
MEDTYP   6-35#
MESSAG   7-16#    46-38
MICREV   6-12#
MOD     35-25    35-51#
MODE    35-52#    45-5*    52-8*
MODE1   35-53#    45-5
MODE2   35-54#    52-8
MRD      5-17#    23-17
MS.CLR  41-110#   44-26
MS.CNT   4-64#    54-26
MS.DIS  41-112#   55-3
MS.END   4-63#    54-10    54-14    54-22    54-29
MS.GCR  41-114#   43-6
MS.GST  41-118#   44-5     46-22
MS.INR  41-124#   46-5
MS.MOD  41-120#   45-6     52-9
MS.ONL  41-108#   43-29
MS.RUN  41-126#   45-19
MS.SCR  41-116#   47-13
MS.SEK  33-16     41-122#
MS.STR   4-62#    54-5     54-19
MS1     25-23     41-8#
MS10    33-29     34-60    41-37#   57-35    61-31    62-19
MS11    41-39#    55-17
MS12    33-33     34-64    41-42#   55-20    57-38    61-34    62-22
MS13    22-8      41-44#
MS14    41-46#    49-46
MS15    41-49#    49-53
MS16    41-52#    46-44
MS17    41-54#    48-37
MS18    41-56#    57-43
MS19    41-58#    57-56
MS2     25-43     41-11#
MS20    41-61#    46-29
MS2000  40-38     41-128#
MS21    41-63#    46-38
MS22    41-65#    61-41
MS23    41-67#    61-57
MS24    34-70     41-71#   62-29
MS25    41-73#    62-40
MS26    41-77#    63-15
MS27    33-41     41-83#
MS28    34-89     41-86#
MS29    41-89#    55-28
MS3     25-47     41-14#
MS30    41-92#    56-33
MS31    41-94#    56-31
MS32    41-96#    56-35
MS33    41-98#    52-94
MS34    29-23     41-100#
MS35    41-102#   51-16
MS4     25-51     41-17#
MS5     25-55     41-20#
MS6     25-59     41-23#
```

```
OUT.15  24-20#
OUT.16  24-21#
OUT.17  24-22#
OUT.18  24-23#
OUT.19  24-24#
OUT.20  24-25#
OUT.21  24-26#
OUT.22  24-27#
OUT.23  24-28#
OUT.24  24-29#
OUT.25  24-30#
OUT.26  24-31#
OUT.27  24-32#
OUT.28  24-33#
OUT.29  24-34#
OUT.30  24-35#
OUT.31  24-36#
OUT.32  24-37#
OUT.33  24-38#
OUT.34  24-39#
OUT.RQ  22-8*    23-14*   23-15    23-20    23-23*   24-5#    24-40    25-23*   25-43*   25-47*   25-51*   25-55*   25-59*   25-64*
        25-71*   29-23*   30-5     30-10    33-29*   33-33*   33-41*   33-42    34-60*   34-64*   34-70*   34-89*   34-90    40-38*
        40-39    42-41*   46-29*   46-44*   48-37*   49-46*   49-53*   51-16*   52-94*   55-17*   55-20*   55-28*   56-31*   56-33*
        56-35*   57-35*   57-38*   57-43*   57-56*   57-57    61-31*   61-34*   61-41*   61-57*   61-58    62-19*   62-22*   62-29*
        62-40*   62-41    63-15*   63-16
OV...    3-3      3-3      3-3#     41-5     41-5     41-5     41-5#    42-2     42-2     42-2     42-2#    50-2     50-2     50-2
        50-2#    63-28    63-28#
OVE.MN   3-3#    41-5
OVE.MS  41-5#    42-2
OVE.SU  42-2#    50-2
OVE.TS  50-2#    63-28
OVERFL   6-75#
OVL...   3-3      3-3#     41-5     41-5#    42-2     42-2#    50-2     50-2#    63-28    63-28#
OVL.MN   3-3      3-3      41-5     41-5     41-5#
OVL.MS  27-25    41-184#  42-2     42-2     42-2#
OVL.SU  27-26    49-105#  50-2     50-2     50-2#
OVL.TS  27-27    63-26#   63-28    63-28    63-28#
OVRLAY  28-33    29-7#    49-103
OVRLNM  29-4     29-27*   29-31#   37-27*
OVRLPT  29-16    29-28    40-56#   42-2     42-2     50-2     50-2
OVRPNT  29-7*    29-23    29-32#
OVS.MN   3-3#
OVS.MS  41-5#
OVS.SU  42-2#
OVS.TS  50-2#
OVSTRT   4-10#   37-29    37-30
PAT4    36-3     36-10#
PAT5    36-4     36-27#
PAT6    36-5     36-44#
PAT8    36-6     36-61#
PATPTR  36-2#    59-9     59-22
PHYSA    3-3      3-3      3-3      3-3      3-3#
PORT2   28-11#   28-41
PORT3   28-27#   28-31
PORT4   28-28    28-32#
PORT5   28-16    28-18    28-20    28-39#
```

```
RBNTRK    6-31#   48-5
RBUF0     34-16   34-53   34-80   34-89   34-89   34-89   36-84#
RBUFD     36-86   37-7#   62-11   63-8
RBUFLN     4-76#
RCONT      4-41#
RCTCPS     6-8#
RCTCSZ     6-40#
RCV        5-8#    25-34   37-93
RCVERR     7-25#   22-32   33-27   34-58   55-15   57-33   61-29   62-17
RCVRDY     7-21#   37-61   42-37
RDSTAT    22-15    22-20#  33-24   34-55   37-52   37-60   38-13   55-12   57-30   61-26   62-14
READ1     34-14#   49-95   50-67   50-75
READ1A    34-53#   34-88
READ1B    34-84    34-88#
READ1C    34-69    34-73#
READ1E    34-79    34-89#
READ1X    34-62    34-66   34-72   34-77   34-92#
READB     59-14    62-2#
REDERR    62-30    62-41#
REDEXT    62-28    62-37   62-39   62-44#
RETS       6-7#
REVECT     6-63#
REVS       6-16#
RGO       62-26    62-31#
RM         6-32#
ROFDBN    35-76#   49-76*  49-77*
ROFDC     35-77#   49-48*  49-49*  49-60*  49-62*  49-68*  52-60*  52-62*  52-64*
RREAL      4-44#   36-88   62-9
RSTOP      4-40#   36-85   62-6
RTDS      22-5#    42-31   44-10   46-13
RTDSL     22-5     22-13#
RUN       35-29    35-45#  52-81*  53-5*   53-13*  53-15*  54-5*   54-8*   54-10*  54-14*  54-19*  54-22*  54-26*  54-27*
          54-29*   55-33*  56-4*   56-29*  56-33
RUNCMD    35-56#   53-15   55-33   56-29
RUNLBC    35-57#   54-27   56-4
RW.ANG     4-34#
RW.BUF     4-29#   61-24*  62-12*
RW.CMD     4-32#   34-89   61-22*  62-10*
RW.HI      4-31#   34-89   61-19*
RW.LOW     4-30#   34-16   34-80   34-89   61-9*
RW.SDI     4-33#
RW.STA     4-28#   61-7*   62-7*
RWRDY      7-26#   33-37   34-68   46-40   57-41   61-37   62-25
SBCRES     7-44#
SCR       35-23    35-42#
SCTWRD     8-8#
SDI       22-14    25-8    26-19   28-32*  30-8    32-9    33-11   33-23   34-54   34-73   35-13#  42-18   52-84   55-11
          56-6     57-29   57-52   61-25   61-44   62-13   62-31
SDILTO    25-30    25-78#  43-18*  52-75*
SDISTO    25-28    25-77#  43-4*   43-14*
SDIVER     6-4#
SECCNT    35-95#   58-13*  58-25   61-39   61-55   62-27   62-38   63-13
SECCYL    35-91#   49-19*  49-20*  50-55
SECGRP    35-90#   49-14*  49-20   49-35   49-71   50-54
SECTRK    31-17    31-31   35-82#  48-10*  49-10   49-94   49-97   50-53   50-66   50-72   52-41   57-17   58-11
SEEK      33-10#   47-33   49-92   50-64   50-73   51-10   52-31   52-67
```

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

```
VAR1     31-38*   31-40*   31-47    35-107#
VAR2     31-42*   31-44*   35-108#
VAR3     31-45*   35-109#
VAR4     31-46*   35-110#
WAITSI    5-13#   34-74    61-45    62-32
WBUFLN    4-75#    4-76
WCONT     4-39#
WFLAG    35-101#  51-2*    51-19*   61-51*
WGO      61-38    61-44#
WREAL     4-43#   61-21
WRITEB   51-11    59-11    61-2#
WRONG     6-65#
WRTEXT   61-40    61-43    61-52    61-54    61-56    61-61#
WS1OP     4-38#   61-6
WTRCMP   58-15    59-2#    59-23
WTREXT   59-13    59-16    59-19    59-24#
XBNCYL    6-41#   48-25
XFERRT    6-5#
XREAD     5-5#    34-75    62-34
XWRITE    5-6#    61-49
```

UDAT3 DISK FUNCTIONAL DMACR X04.01 13-APR-82 18:08:21 PAGE M-1
CROSS REFERENCE TABLE (CREF V04.00)

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22
TABLE OF CONTENTS

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22
TABLE OF CONTENTS

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22
TABLE OF CONTENTS

```
                                .TITLE  UDAT4   DISK EXERCISER
                        ;
                        ;
                        ; COPYRIGHT (C)  1981
                        ; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.
                        ;
                        ;
                        ; THIS  SOFTWARE  IS  FURNISHED  UNDER A LICENSE FOR USE ONLY ON A
                        ; SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION
                        ; OF  THE  ABOVE COPYRIGHT  NOTICE.  THIS SOFTWARE, OR  ANY  OTHER
                        ; COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE  MADE AVAILABLE
                        ; TO  ANY  OTHER PERSON  EXCEPT  FOR USE ON SUCH SYSTEM AND TO ONE
                        ; WHO AGREES TO THESE LICENSE TERMS.  TITLE  TO  AND  OWNERSHIP OF
                        ; THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DIGITAL.
                        ;
                        ; THE  INFORMATION  IN THIS DOCUMENT IS SUBJECT  TO CHANGE WITHOUT
                        ; NOTICE AND SHOULD NOT BE CONSTRUED AS A  COMMITMENT  BY  DIGITAL
                        ; EQUIPMENT CORPORATION.
                        ;
                        ; DIGITAL ASSUMES NO  RESPONSIBILITY FOR THE USE OR RELIABILITY OF
                        ; ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;THIS PROGRAM SHALL BE ASSEMBLED WITH THE PROGRAM DMACRO
                        ;USING A COMMAND LINE SIMILAR TO:
                        ;
                        ;       UDAT4.BIN,UDAT4/C=$DMACRO,UDAT4
                        ;
        000000          DEBUG   =       0                       ; SET TO 1 TO DEBUG TEST4
                        ;       .ENABL  ABS
000000                  ;       DMCODE  UDADM4,0,714,3,0,34200
```

```
                                .SBTTL  MODIFICATION CONTROL
000000                                  .REPT   0


                                        MODIFICATION CONTROL -- ALL CHANGES AFTER OCTOBER 29, 1981 WILL BE
                                        ENTERED IN THIS SECTION TO ALLOW THE HISTORY OF TEST 4 TO BE DOCUMENTED
                                        .ENDR
```

```
                        .SBTTL  UDA DM PROGRAM PARAMETERS
                        .LIST MEB
                        ;
                        ;       EQUATES
                        ;
                        ;       HIGHEST USABLE LOCATION OF UDA MEMORY + 1
                        ;
007774                  HIMEM   =       7774            ; HIGH MEMORY+1
007774                  OVSTRT  =       7774            ; OVERLAY ADDRESS LOCATION
                        ;
                        ;
                        ;       OFFSETS FOR FORMAT TRACK TABLE
                        ;
000000                  FT.BUF  =       0.              ; BUFFER POINTER OFFSET
000001                  FT.HI   =       1.              ; HI ORDER HEADER OFFSET
000002                  FT.LOW  =       2.              ; LOW ORDER HEADER OFFSET
                        ;
                        ;
                        ;       OFFSETS FOR FORMAT TRACK BUFFER
                        ;
000000                  FB.DAT  =         0.            ; FIRST DATA WORD OFFSET
000400                  FB.EDC  =       256.            ; EDC WORD OFFSET
                        ;
                        ;
                        ;       OFFSETS FOR READ/WRITE I/O CHAIN TABLES
                        ;
000000                  RW.STAT =       0.              ; STATUS AND NEXT BUFFER POINTER OFFSET
000001                  RW.BUF  =       1.              ; POINTER TO DATA BUFFER
000002                  RW.LOW  =       2.              ; HI ORDER EXPECTED HEADER
000003                  RW.HI   =       3.              ; LOW ORDER EXPECTED HEADER
000004                  RW.CMD  =       4.              ; SDI COMMAND AND HEAD ADDRESS
000005                  RW.SDI  =       5.              ; DUMMY SDI CONTROL BLOCK POINTER
000006                  RW.ANG  =       6.              ; THETA FROM INDEX
                        ;
                        ;       CONSTANTS FOR READ AND WRITE XFC'S
                        ;
140000                  WSTOP   =       140000          ; LAST ENTRY IN CHAIN FOR WRITE
040000                  WCONT   =       40000           ; WRITE CONTINUE
100000                  RSTOP   =       100000          ; LAST ENTRY IN CHAIN FOR READ
000000                  RCONT   =       0               ; READ CONTINUE
100000                  FSTOP   =       100000          ; LAST ENTRY IN CHAIN FOR FORMAT
122400                  WREAL   =       122400          ; WRITE REAL TIME ECOMMAND
013400                  RREAL   =       13400           ; READ REAL TIME COMMAND
010000                  ECCFLG  =       10000           ; ECC ERROR IN BUFFER BIT
100000                  EOC     =       100000          ; END OF CHAIN
040000                  BUFFLG  =       40000           ; BUFFER FULL OR EMPTY FLAG
                        ;
                        ;
                        ;       HEADER CODES
                        ;
000000                  HD.LBN  =       000000          ; GOOD LBN
060000                  HD.RBN  =       060000          ; GOOD RBN, PERHAPS UNUSED
030000                  HD.REV  =       030000          ; REVECTORED LBN
110000                  HD.BAD  =       110000          ; BAD BLOCK
050000                  HD.PRV  =       050000          ; PRIMARY REVECTORED BLOCK
120000                  HD.XBN  =       120000          ; XBN BLOCK
```

UDA DM PROGRAM PARAMETERS

```
58          140000                  HD.DBN  =       140000          ; DBN BLOCK
59
60                                  ;       OFFSETS FOR DATA BUFFERS
61
62          000000                  BF.DAT  =         0.            ; DATA
63          000400                  BF.EDC  =       256.            ; ERROR DETECTION CODE
64          000401                  BF.ECC  =       257.            ; LAST 12 ECC RESIDUES
65
66                                  ;       BUFFER AND READ/WRITE CHAIN LINK SIZES
67
68          000401                  WBUFLN  =       257.            ; WRITE BUFFER SIZE
69          000415                  RBUFLN  =       WBUFLN+12.      ; READ BUFFER SIZE
70          000007                  LINKLN  =       7.              ; LINK SIZE
```

```
                                ;       XFC DEFINITION EQUATES
                                ;
000000                          BREAK   =       0.              ; BREAKPOINT XFC CODE
000001                          FORMAT  =       1.              ; FORMAT TRACK XFC CODE
000002                          XREAD   =       2.              ; READ N SECTORS XFC CODE
000003                          XWRITE  =       3.              ; WRITE N SECTORS XFC CODE
000004                          SEND    =       4.              ; SEND SDI COMMAND XFC CODE
000005                          RCV     =       5.              ; RECEIVE SDI MESSAGE XFC CODE
000006                          COMPARE =       6.              ; COMPARE DATA PATTERN TO BUFFER
000007                          STATUS  =       7.              ; RETURN DRIVE STATUS XFC CODE
000010                          ECHO    =       8.              ; ECHO DATA TO DRIVE XFC CODE
000011                          DINIT   =       9.              ; DRIVE INITIALIZE XFC CODE
000012                          WAITSI  =       10.             ; WAIT FOR SECTOR OR INDEX PULSE
000013                          UREAD   =       11.             ; READ UNIBUS MEMORY XFC CODE
000014                          UWRITE  =       12.             ; WRITE UNIBUS MEMORY XFC CODE
000015                          ECC     =       13.             ; DO ECC ON BUFFER XFC CODE
000016                          MRD     =       14.             ; SEND BUFFER TO MAINTENANCE READ COMMAND
000017                          MWR     =       15.             ; GET BUFFER FROM MAINTENANCE WRITE COMMAND
000020                          CVT     =       16.             ; CONVERT TO PHYSICAL ADDRESS XFC CODE
000021                          EXIT    =       17.             ; TERMINATE DM PROGRAM XFC CODE
                                ;
                                ;       MEDIA TYPES
                                ;
                                ;       THIS WORD IS FOUND AS THE FIRST WORD OF XBN 0 (OR ANY OF ITS COPIES)
                                ;
126736                          MOD512  =       126736          ; 512 BYTE MODE
074161                          MOD576  =       074161          ; 576 BYTE MODE
                                ;
                                ;       GET STATUS OFFSETS
                                ;
000000                          ST.UNT  =       0.              ; UNIT NUMBER
000000                          ST.MSK  =       0.              ; SUBUNIT MASK
000001                          ST.REQ  =       1.              ; REQUEST BYTE
000001                          ST.MOD  =       1.              ; MODE BYTE
000002                          ST.ERR  =       2.              ; ERROR BYTE
000002                          ST.CON  =       2.              ; CONTROLLER BYTE
000002                          ST.C    =       2.              ; C BITS
000003                          ST.RTY  =       3.              ; RETRY COUNT/FAILURE CODE
100000                          VALID   =       100000          ; USED IN ERROR POSITION TO MARK VALID STATUS
                                                                ;  IN INPUT BUFFER
                                ;
                                ;       STATUS BIT DEFINITIONS
                                ;
000200                          ST.OA   =       200             ; ONLINE TO ANOTHER (SET IF DRIVE UNAVAILABLE)
000100                          ST.RR   =       100             ; READJUSTMENT BIT (SET IF RECALIBRATION REQUIRED)
000040                          ST.DR   =       40              ; DIAGNOSTIC REQUEST (SET IF DIAGNOSTIC REQUESTED)
000020                          ST.SR   =       20              ; SPINDLE READY (SET IF SPINDLE READY)
000010                          ST.EL   =       10              ; LOGGABLE INFO IN EXTENDED STATUS
000002                          ST.PS   =       2               ; PORT SWITCH (SET IF PORT SWITCH IN)
000001                          ST.RU   =       1               ; RUN/STOP SWITCH (SET IF RUN/STOP SWITCH IN)
000200                          ST.DE   =       200             ; DRIVE ERROR
000100                          ST.RE   =       100             ; RETRIABLE ERROR (SET IF RETRIABLE ERROR OCCURRED)
000040                          ST.PE   =       40              ; PROTOCOL ERROR (SET IF PROTOCOL ERROR OCCURRED)
000020                          ST.DF   =       20              ; INITIALIZATION FAILURE (SET IF INIT FAILED)
000010                          ST.WE   =       10              ; WRITE ENABLE (SET IF WRT ATTEMPTED ON PROT DISK)
002000                          ST.FO   =       2000            ; FORMATTING (SET IF FORMATTING ENABLED)
001000                          ST.DB   =       1000            ; DIAGNOSTIC CYLS (SET IF DIAG CYL ACCESS ENABLED)
```

```
     58          000400                    ST.S7   =       400           ; SECTOR SIZE (SET FOR 576 BYTE SECTORS)
```

```
                                ;       GET COMMON CHARACTERISTICS OFFSETS
                                ;
000000                          SHRTTO  =       0.              ; SHORT TIMEOUT <3:0>
000000                          SDIVER  =       0.              ; SDI VERSION <7:4>
000000                          XFERRT  =       0.              ; TRANSFER RATE <15:0>
000001                          LONGTO  =       1.              ; LONG TIMEOUT <3:0>
000001                          RETS    =       1.              ; RETRIES <7:4>
000001                          RCTCPS  =       1.              ; F/RCT COPIES <11:8>
000001                          SS      =       1.              ; SECTOR SIZE <15:15>
000002                          ERLEV   =       2.              ; ERROR RETRY LEVELS <7:0>
000002                          ECCRSH  =       2.              ; ECC THRESHOLD <15:8>
000003                          MICREV  =       3.              ; MICROCODE REVISION NUMBER <7:0>
000003                          HRDREV  =       3.              ; HARDWARE REVISION NUMBER <15:8>
000004                          DRVID   =       4.              ; UNIQUE DRIVE ID <47:0>
000007                          DRTYPE  =       7.              ; DRIVE TYPE IDENTIFIER <7:0>
000007                          REVS    =       7.              ; REVS/SECOND <15:8>
                                ;
                                ;       GET SUBUNIT CHARACTERISTICS OFFSETS
                                ;
                                ;THESE OFFSETS ARE CURRENTLY GIVEN AS FOLLOWING THE COMMON CHARACTERISTICS
                                ;
000000                          LBNCYL  =       0.              ; NUMBER OF CYLINDERS IN LBN AREA <31:0>
000001                          HICYL   =       1.              ; HI ORDER CYLINDER BITS <15:12>
000002                          GRPCYL  =       2.              ; GROUPS PER CYLINDER <7:0>
000002                          HILBN   =       2.              ; HI STARTING LBN <11:8>
000002                          HIXBN   =       2.              ; HI STARTING XBN <15:12>
000003                          TRKGRP  =       3.              ; TRACKS PER GROUP <7:0>
000003                          HIRBN   =       3.              ; HI STARTING RBN <11:8>
000003                          HIDBN   =       3.              ; HI STARTING DBN <15:12>
000004                          RBNTRK  =       4.              ; RBNS PER TRACK <6:0>
000004                          RM      =       4.              ; REMOVABLE MEDIA <7:7> 1=REMOVEABLE
000005                          DATPRE  =       5.              ; DATA PREAMBLE SIZE IN WORDS <7:0>
000005                          HDRPRE  =       5.              ; HEADER PREAMBLE SIZE IN WORDS <15:8>
000006                          MEDTYP  =       6.              ; MEDIA TYPE <31:0>
000010                          FCTSIZ  =       8.              ; FCT COPY SIZE <15:0>
000011                          LBNTRK  =       9.              ; LBNS PER TRACK <7:0>
000011                          GRPOFF  =       9.              ; GROUP OFFSET (SECTORS) <15:8>
000012                          LBNHST  =       10.             ; LBNS IN HOST AREA <31:0>
000014                          RCTCSZ  =       12.             ; RCT COPY SIZE <15:0>
000021                          XBNCYL  =       17.             ; CYLS IN XBN AREA <15:0>
000022                          DBNCYL  =       18.             ; CYLS IN DBN AREA <15:8>
                                ;
                                ;
                                ;       BIT MASK DEFINITIONS
                                ;
                                ;
177400                          HIBYTE  =       177400          ; HIGH BYTE MASK
000377                          LOBYTE  =       000377          ; LOW BYTE MASK
007777                          HBHINB  =       7777            ; HI BYTE, HI NIBBLE MASK
170377                          HBLONB  =       170377          ; HI BYTE, LO NIBBLE MASK
177417                          LBHINB  =       177417          ; LO BYTE, HI NIBBLE MASK
177760                          LBLONB  =       177760          ; LO BYTE, LO NIBBLE MASK
                                ;
```

```
                                ;MAINTANENCE READ/WRITE REQUEST NUMBERS
                                ;
060000                          DUPPKT  =       060000          ; DUP SPECIAL PACKET NUMBER
060000                          T1MSIZ  =       0.+DUPPKT       ; GET FREE MEMORY PARAMETERS
060001                          T2DLL   =       1.+DUPPKT       ; DOWNLINE LOAD DRIVE DIAGNOSTIC
060002                          T2CMD   =       2.+DUPPKT       ; MANUAL INTERVENTION TEST 2 PROTOCOL
060003                          T4MPRM  =       3.+DUPPKT       ; GET MASTER PARAMETERS FROM SW QUESTIONS
060004                          T4UPRM  =       4.+DUPPKT       ; GET UNIT PARAMETERS FROM HW QUESTIONS
060005                          T4BB1   =       5.+DUPPKT       ; GET BAD BLOCKS (1 THRU 14)
060006                          T4BB2   =       6.+DUPPKT       ; GET REST OF BAD BLOCKS (15 AND 16)
060007                          T4SOFT  =       7.+DUPPKT       ; ADD TO SOFT ERROR AND ECC COUNT
060010                          T4SEEK  =       8.+DUPPKT       ; ADD 1 TO SEEK COUNT
060011                          T4MXFR  =       9.+DUPPKT       ; ADD TO MEGABITS READ AND WRITTEN
060012                          UTOTST  =       10.+DUPPKT      ; GET UNITS TO TEST
060013                          ERRMES  =       11.+DUPPKT      ; PRINT ERROR MESSAGE
060014                          ERRMC   =       12.+DUPPKT      ; TEST 4 ERROR REPORTING
060015                          MESSAG  =       13.+DUPPKT      ; INFORMATION MESSAGE
060016                          DONE    =       14.+DUPPKT      ; MARK DM PROGRAM AS NO LONGER RUNNING
                                ;
                                ;       STATE BIT DEFINITIIONS
                                ;
000001                          RCVRDY  =       1               ; RECIEVER READY 1 = READY
000002                          ATTN    =       2               ; ATTENTION BIT FOR RETURN DRIVE SIGNALS XFC
000004                          DCLOCK  =       4               ; CONTROLLER STATE READY -- THERE ARE CLOCKS
                                                                ; COMING FROM THE DRIVE
000100                          AVAIL   =       100             ; AVAILABLE 1 = AVAILABLE
000400                          RCVERR  =       400             ; RECEIVE ERROR -- PARITY ERROR
100000                          RWRDY   =       100000          ; IF SET, UDA IS ABLE TO READ AND/OR WRITE TO DRIVE
                                ;
                                ;       SDI COMMANDS AND RESPONSES
                                ;
000204                          DISCON  =       204             ; DISCONNECT DRIVE
000006                          ERECOV  =       6               ; ERROR RECOVERY
000201                          CHGMOD  =       201             ; CHANGE MODE
000213                          DRVONL  =       213             ; DRIVE ONLINE
000014                          DRVRUN  =       14              ; DRIVE RUN
000005                          DRVCLR  =       5               ; DRIVE CLEAR OPCODE
000207                          GETCHR  =       207             ; GET CHARACTERISTICS
000210                          GETSUB  =       210             ; GET SUBUNIT CHARACTERISTICS
000011                          GETSTA  =       11              ; GET STATUS
000216                          IRECLB  =       216             ; RECALIBRATE
000012                          INSEEK  =       12              ; INITIATE SEEK
000176                          COMPLT  =       176             ; SUCCESSFUL COMPLETION
000175                          UNSSUC  =       175             ; UNSUCCESSFUL COMPLETION
000170                          CHRRES  =       170             ; GET CHARACTERISTICS RESPONSE
000167                          SBCRES  =       167             ; GET SUBUNIT CHARACTERISTICS RESPONSE
000366                          STSRES  =       366             ; GET STATUS RESPONSE
000350                          ECHOC   =       350             ; DIAGNOSTIC ECHO COMMAND AND RESPONS:
                                ;
                                ;       SDI LEVEL 2 COMMAND OFFSETS
                                ;
000000                          L2.OPC  =       0               ; SDI LEVEL 2 OP CODE
000001                          L2.SLN  =       L2.OPC+1        ; SDI LEVEL 2 SEND LENGTH IN BYTES
000002                          L2.RLN  =       L2.SLN+1        ; SDI LEVEL 2 RECEIVE LENGTH IN WORDS
000003                          L2.ERS  =       L2.RLN+1        ; SDI LEVEL 2 EXPECTED RESPONSE
000004                          L2.EOF  =       L2.ERS+1        ; SDI LEVEL 2 ERROR OFFSET INTO UNIT PARAMETERS
                                ;
```

```
                                ;       ERROR CODES
                                ;
000000                          FTLSYS  =       0               ; SYSTEM FATAL ERROR
040000                          FTLDEV  =       040000          ; DEVICE FATAL
100000                          ERHARD  =       100000          ; HARD ERROR
140000                          ERSOFT  =       140000          ; SOFT ERROR
040000                          C2HARD  =       <ERHARD&^CERSOFT>!<ERSOFT&^CERHARD>     ; CHANGE SOFT TO HARD ERROR
100000                          C2DFTL  =       <FTLDEV&^CERSOFT>!<ERSOFT&^CFTLDEV>     ; CHANGE SOFT TO DEVICE FATAL
```

```
                        .SBTTL  TEST 4 SPECIFIC INFORMATION
                        ;
                        ;       TEST 4 SPECIFIC INFORMATION
                        ;
                        ;
                        ;
                        ;       UNIT PARAMETER OFFSETS
                        ;
                        ;
000000                  U.NEXT  =       0.              ; POINTER TO NEXT UNIT (RING LINKED LIST)
000001                  U.SUBP  =       U.NEXT+1        ; 4 WORDS OF SUBUNIT PARAMETER POINTERS
000005                  U.TIMH  =       U.SUBP+4        ; HI ORDER TIMEOUT FOR SEEK
000006                  U.TIML  =       U.TIMH+1        ; LO ORDER TIMEOUT FOR SEEK
000007                  U.SRTY  =       U.TIML+1        ; ISSUE SEEK TIMEOUT COUNTER
000010                  U.RRTY  =       U.SRTY+1        ; READ RETRIES
000011                  U.MSTO  =       U.RRTY+1        ; MASTER SEEK TIMEOUT
000012                  U.RWTO  =       U.MSTO+1        ; READ/WRITE TIMEOUT AREA
000013                  U.NFUN  =       U.RWTO+1        ; NEXT FUNCTION ADDRESS (FOR DEFERRED CALLS)
000014                  U.PAT   =       U.NFUN+1        ; PATTERN NUMBER TO WRITE
000015                  U.CCNT  =       U.PAT+1         ; CURRENT COUNT OF T/G LOOPS
000017                  U.PCTG  =       U.CCNT+2        ; POINTER TO CURRENT TRACK OR GROUP
000020                  U.CTRK  =       U.PCTG+1        ; TRACK COUNT FOR GROUP OPERATIONS
000021                  U.NSEC  =       U.CTRK+1        ; NUMBER OF SECTORS R/W THIS TRY
000022                  U.MSEC  =       U.NSEC+1        ; NUMBER OF SECTORS TO BE R/W
000023                  U.TSEC  =       U.MSEC+1        ; NUMBER OF SECTORS TO BE R/W THIS OP
000024                  U.CSEC  =       U.TSEC+1        ; COUNT OF SECTORS R/W SO FAR
000025                  U.MASK  =       U.CSEC+1        ; UNIT MASK FOR XFC CALLS (0001 - 1000)
000026                  U.WRIT  =       U.MASK+1        ; WRITE PROTECTION STATUS
000027                  U.ELEV  =       U.WRIT+1        ; CURRENT ERROR RECOVERY LEVEL
000030                  U.RTRY  =       U.ELEV+1        ; MAXIMUM NUMBER OF READ RETRIES
000031                  U.MLEV  =       U.RTRY+1        ; MAXIMUM NUMBER OF ERROR RECOVERY LEVELS
000032                  U.ECCT  =       U.MLEV+1        ; ECC THRESHOLD
000033                  U.SDIS  =       U.ECCT+1        ; SDI SHORT TIMEOUT
000034                  U.SDIL  =       U.SDIS+1        ; SDI LONG TIMEOUT
000035                  U.SDI2  =       U.SDIL+1        ; LEVEL 2 ERROR COUNTS
000045                  U.WPRT  =       U.SDI2+NUML2S   ; MASK TO WRITE PROTECT READ-ONLY DRIVES
000046                  U.PARM  =       U.WPRT+1        ; UNIT PARAMETER WORD
000047                  U.RCOV  =       U.PARM+1        ; RECOVERY PARAMETERS
000050                  U.SUBU  =       U.RCOV+1        ; SUBUNIT OFFSET (0 - 3)
000051                  U.MBN   =       U.SUBU+1        ; MASTER L/DBN
000053                  U.CBN   =       U.MBN+2         ; CURRENT L/DBN FOR START OF CHAIN
000055                  U.RBN   =       U.CBN+2         ; RBN TO BE READ/WRITTEN IF LBN REVECTORED
000057                  U.COPY  =       U.RBN+2         ; NUMBER OF RCT COPIES ON EACH SUBUNIT
000060                  U.CCOP  =       U.COPY+1        ; CURRENT RCT COPY THAT WE'RE WORKING ON
000061                  U.RWER  =       U.CCOP+1        ; ERROR (IF ANY) ON THE LAST R/W
000062                  U.RVER  =       U.RWER+1        ; ERROR THAT OCCURRED BEFOR THE REVECTOR OPERATION
000063                  U.UNUM  =       U.RVER+1        ; STARTING SUBUNIT NUMBER
000064                  U.CCYL  =       U.UNUM+1        ; CURRENT CYLINDER
000066                  U.CGRP  =       U.CCYL+2        ; CURRENT GROUP
000067                  U.LCYL  =       U.CGRP+1        ; LAST CYLINDER SEEKED TO
000071                  U.LGRP  =       U.LCYL+2        ; LAST GROUP SEEKED TO
                        ;
                        ;       SUBUNIT PARAMETER OFFSET
                        ;
000000                  S.PARM  =       0.              ; SUBUNIT PARAMETER WORD
000001                  S.SEEK  =       S.PARM+1        ; COUNT OF NUMBER OF SEEKS
000002                  S.SDCL  =       S.SEEK+1        ; STARTING DIAGNOSTIC CYLINDER
```

```
                                ;
                                ;       IF TRACK/GROUP LIMITS ARE GIVEN, THE SUBUNIT PARAMETERS HAVE THE
                                ;       FOLLOWING FIELDS ADDED TO THEM
                                ;
000013          S.MCNT  =       S.BESS          ; MAXIMUM TRACK/GROUP COUNT
000015          S.TGOF  =       S.MCNT+2        ; ORIGINAL TRACK/GROUP OFFSET
00001?          S.TGSS  =       S.TGOF+2        ; START OF TRACK/GROUP SETS
                                ;
                                ;       CONSTANTS
                                ;
000377          SCTWRD  =       255.            ; NUMBER OF WORDS IN SECTOR TO FILL
000105          INTEDC  =       69.             ; INITIAL EDC VALUE
000072          TLEN.U  =       U.LGRP+1        ; UNIT PARAMETER LENGTH
007702          FIRSTU  =       HIMEM-TLEN.U    ; LOCATION OF FIRST UNIT PARAMETER BLOCK
177774          MAXMSK  =       177774          ; TO DETERMINE MAXIMUM SECTORS TO READ/WRITE
001750          MAXSND  =       1000.           ; MAXIMUM TIMES TO SEND A COMMAND TO AN OFFLINE DRIVE
```

```
                                ;
                                ;       DUMMY SDI CONTROL BLOCK OFFSETS
                                ;
000001                          D.LIMT  =       1               ; DUMMY SDI SEARCH LIMIT
000002                          D.SCHR  =       2               ; DUMMY POINTER TO SUBUNIT CHAR-5
                                ;
                                ;       MASTER PARAMETER BITS (M.PARM)
                                ;
100000                          IWIPRG  =       100000          ; INITIAL WRITE IN PROGRESS
000002                          DIE     =       000002          ; ERRORS DURING INITIALIZATION
000001                          INTINP  =       000001          ; INITIAL SETUP IN PROGRESS
                                ;
                                ;       UNIT PARAMETER BITS (U.PARM(R5))
                                ;
100000                          DROP    =       100000          ; DROP BIT (SET IF UNIT OR SUBUNIT DROPPED)
040000                          INITW   =       40000           ; INITIAL WRITE (SET IF INITIAL WRITE IN PROG)
010000                          DIREC   =       10000           ; DIRECTION (SET IF          AL ACCESSES DECREASING)
004000                          NEWSUB  =       4000            ; SET IF          AL SEEKS MOVED TO NEW SUBUNIT
002000                          SEKINP  =       2000            ; SEEK IN PROGRESS - SET IF TRUE
001000                          FTIME   =       1000            ; FIRST TIME FLAG - SET FOR INIT CODE
000400                          REVEC   =       400             ; REVECTOR BIT (SET IF BLOCK REVECTORED)
000200                          RBNBN   =       200             ; SEE IF WORKING ON RBN
000100                          REDWRT  =       100             ; REDWRT (READ OR WRITE IN PROGRESS SET IF WRITE)
                                ;               10              ; RESERVED (SEE SUBUNIT PARAMETER BITS)
                                ;               2               ; RESERVED (SEE SUBUNIT PARAMETER BITS)
                                ;
                                ;       ERROR RECOVERY PARAMETERS (U.RCOV)
                                ;
100000                          ERMASK  =       100000          ; IF 1, ERROR RECOVERY IS DISABLED
040000                          DRINIT  =       40000           ; IF 1, INDICATES THAT DRIVE WAS INITIALIZED
020000                          LEVUSD  =       20000           ; IF 1, RE-ISSUE THIS LEVEL OF ERROR RECOVERY
010000                          NXTLEV  =       10000           ; IF 1, ISSUE NEW LEVEL OF ERROR RECOVERY
004000                          SEKREQ  =       4000            ; IF 1, SEEK IS REQUIRED FOR ERROR RECOVERY
002000                          RCBREQ  =       2000            ; IF 1, RECALIBRATE IS REQUIRED FOR ERROR RECOVERY
001000                          RETRY   =       1000            ; IF 1, JUST RETRY THE SAME MODULE
                                ;
                                ;       SUBUNIT PARAMETER BITS (S.PARM(R4))
                                ;
                                ;               100000          ; RESERVED (DROP BIT)
                                ;               40000           ; RESERVED (INITIAL WRITE)
020000                          DCYLS   =       20000           ; DIAGNOSTIC CYLINDER FLAG (SET IF DBNS)
010000                          ECCCHK  =       10000           ; 1 IF ECC CORRECTION ALLOWED   MASTER BITS
004000                          RONLY   =       4000            ; READ ONLY (SET IF READ ONLY)
002000                          WONLY   =       2000            ; WRITE ONLY (SET IF WRITE ONLY)
001000                          RTRIES  =       1000            ; 1 IF RETRIES ALLOWED
000200                          ONLYCL  =       200             ; SET IF ONLY CYLINDERS SPECIFIED
                                                                ; USED DURING SETUP ONLY
000100                                          100             ;          AL SEEK (START UP TESTING ONLY)
000040                          BEUSED  =       40              ; BEGIN/END SETS (USED IF SET)
000020                          TRACKS  =       20              ; TRACKS OR GROUPS (TRACK IF SET)
000010                          WCHECK  =       10              ; WRITE CHECK BIT (IF SET, WRITE CHECK WILL BE DONE)
                                ;                       WCHECK IS ALSO USED FOR THE UNIT PARAMETERS
000004                          WCHKAL  =       4               ; SET IF WRITE CHECK ALWAYS TO BE DONE
000002                          DATCMP  =       2               ; DATA COMPARE (SET IF DATA COMPARE TO BE DONE)
```

```
     58                                 ;
     59          000001                 DCMPAL  =       1          DATCMP IS ALSO USED FOR THE UNIT PARAMETERS
                                                                         ; SET IF DATA COMPARE ALWAYS TO BE DONE
```

MACRO DEFINITIONS

```
                        .SBTTL  MACRO DEFINITIONS
                        ;
                        ;       DIAGNOSTIC MACRO FOR TEST4 OVERLAYS
                        ;
                                .MACRO  DIAG$$
                                TST     $$DIAG+$DIAG$
                                BEQ     .+6
                                MOV     #60000,R0
                                MOV     R0,@$$DIAG+$DIAG$
                                BR      .+1
                        $DIAG$  =       $DIAG$ + 1
                                .ENDM
```

MACRO DEFINITIONS

```
                                ;       OVERLAY INFORMATION MACRO
                                ;
                                        .MACRO  DFOVLY  LABL$,ONAME,SAREA,EAREA
                                        .WORD   <SAREA-PART0>/2
                                        .IF     NB,EAREA
                                        .WORD   <EAREA-SAREA>/2
                                        .IFF
                                        .WORD   0
                                        .ENDC
                                        .WORD   0
                                        .WORD   OVL.'ONAME'*4
                                        .IF     NE,OCNT$-LABL$
                                        .ERROR  ; OVERLAY NUMBER AND POSITION IN TABLE DO NOT MATCH
                                        .ENDC
                                OCNT$   =       OCNT$+1
                                        .ENDM
                                ;
                                ;       MESSAGE CONTROL TABLE MACRO
                                ;
                                        .MACRO  MSG CMDBUF,CMDSZ,RPLSZ,SUCCOM
                                        .WORD   CMDBUF                  ; ADDRESS OF COMMAND
                                        .WORD   CMDSZ                   ; SIZE OF COMMAND IN BYTES
                                        .WORD   RPLSZ                   ; SIZE OF REPLY IN WORDS
                                        .WORD   SUCCOM                  ; SUCCESSFUL COMPLETION CODE
                                        .WORD   SDI$+U.SDI2             ; RETRY COUNT OFFSET
                                SDI$    =       SDI$+1
                                        .ENDM
```

MACRO DEFINITIONS

```
      1                                .MACRO  BCS     LAB..
      2                                                        BCC     .+2
      3                                                        BR      LAB..
      4                                .ENDM
```

```
                        ;       PUSH REGISTER MACRO
                        ;
                                .MACRO PUSH R9
                                .IRP X,<R9>
                                                                MOV X,-(SP)
                                .ENDR
                                .ENDM
                        ;
                        ;       POP REGISTER MACRO
                        ;
                                .MACRO POP R9
                                .IRP X,<R9>
                                                                MOV (SP)+,X
                                .ENDR
                                .ENDM
```

```
                        ;
                        ;       PRIMARY ERROR REPORTING (TEST 4)
                        ;
                        .MACRO  SOFTER  NUM,ARGS
                        ERROR   ERSOFT,NUM,<ARGS>
                                                                MOV     #ERRMES,HRQ.RQ
                        .ENDM

                        .MACRO  REPSFT  SFTFLG,ECCFG,SEKFLG,COMFLG
                        .IF     NB,SFTFLG
                        .IF     IDN,SFTFLG,SOFT
                                                                MOV     #1,HRQ.02
                        .IFF
                        .ERROR  ; NON SOFT ERROR FLAG IN SOFT ERROR POSITION
                        .ENDC
                        .IFF
                                                                CLR     HRQ.02
                        .ENDC
                        .IF     NB,ECCFG
                        .IF     IDN,ECCFG,ECC
                                                                MOV     #1,HRQ.03
                        .IFF
                        .ERROR  ; NON ECC ERROR FLAG IN ECC ERROR POSITION
                        .ENDC
                        .IFF
                                                                CLR     HRQ.03
                        .ENDC
                        .IF     NB,SEKFLG
                        .IF     IDN,SEKFLG,SEEK
                                                                MOV     #1,HRQ.04
                        .IFF
                        .ERROR  ; NON SEEK ERROR FLAG IN SEEK ERROR POSITION
                        .ENDC
                        .IFF
                                                                CLR     HRQ.04
                        .ENDC
                ;       .IF     NB,COMFLG
                ;       .IF     IDN,COMFLG,COMM
                ;                                               MOV     #1,HRQ.05
                ;       .IFF
                ;       .ERROR  ; NON COMMUMICATION ERROR FLAG IN COMMUNICATION ERROR POSITION
                ;       .ENDC
                ;       .IFF
                ;                                               CLR     HRQ.05
                ;       .ENDC
                                                                PUSH    R0
                                                                MOV     U.UNUM(R5),R0
                                                                ADD     U.SUBU(R5),R0
                                                                MOV     R0,HRQ.01
                                                                MOV     #T4SOFT,R0
                                                                CALL    HOSTRQ
                                                                POP     R0
                        .ENDM

                        .MACRO  HARDER  NUM,ARGS
                        ERROR   ERHARD,NUM,<ARGS>
                                                                MOV     #ERRMC,HRQ.RQ
```

```
                        .ENDM

                        .MACRO  DEVFTL  NUM,ARGS
                        ERROR   FTLDEV,NUM,<ARGS>
                                                        MOV     #ERRMC,HRQ.RQ
                        .ENDM

                        .MACRO  SYSFTL  NUM,ARGS
                        ERROR   FTLSYS,NUM,<ARGS>
                                                        MOV     #ERRMC,HRQ.RQ
                        .ENDM

                        .MACRO  ERROR   TYPE,NUM,ARGS
                        .RADIX  10
                        NUMPTR  =       5
                        MOVMSG  #ER'NUM,4
                        .IF     NB,<ARGS>
                        .IRP    X,<ARGS>
                        MOVMSG  X,\NUMPTR
                        NUMPTR  =       NUMPTR + 1
                        .ENDR
                        .ENDC
                                                        MOV     #NUM!TYPE+4000.,R2
                                                        MOV     R2,HRQ.02
                                                        MOV     #.,HRQ.01
                        .RADIX
                        .ENDM

                        .MACRO  CERROR  STNUM,ARGS
                        .RADIX  10
                NUMPTR  =       STNUM
                        .IRP    X,<ARGS>
                        MOVMSG  X,\NUMPTR
                NUMPTR  =       NUMPTR + 1
                        .ENDR
                        .RADIX
                        .ENDM

                        .MACRO  ERRORC  ARGS
                        .RADIX  10
                        .IRP    X,<ARGS>
                        MOVMSG  X,\NUMPTR
                NUMPTR  =       NUMPTR + 1
                        .ENDR
                        .RADIX
                        .ENDM

                        .MACRO  MOVMSG  ARG,INDX

                        .IF     LT,INDX-10
                                                        MOV     ARG,HRQ.0'INDX
                        .IFF
                                                        MOV     ARG,HRQ.'INDX
                        .ENDC
                        .ENDM

                        .MACRO  ENDERR  POS
```

```
                        .RADIX  10
                        .IF     NB,POS
                        NDERR   POS
                        .IFF
                        NDERR   \NUMPTR
                        .ENDC
                        .RADIX
                        .ENDM

                        .MACRO  NDERR   POS
                        .IF     NE,POS
                        MOVMSG  #SER22,POS
                                                                MOV     #POS+1,ERRPOS   ; SET THE POSITION
                        .IF     LT,26.-POS
                        .ERROR  ; STATUS AT END OF ERROR MESSAGE WILL OVERFLOW HOST COMMUNICATION BUFFER
                        .ENDC
                        .IFF
                                                                CLR     ERRPOS          ; CLEAR THE POSITION
                        .ENDC
                        .ENDM

;
;       MESSAGE REPORTING MACRO
;
                        .MACRO  MSSG    NUM,ARGS
                        .RADIX  10
                        NUMPTR  =       3
                        MOVMSG  #MS'NUM,2
                        .IF     NB,<ARGS>
                        .IRP    X,<ARGS>
                        MOVMSG  X,\NUMPTR
                        NUMPTR  =       NUMPTR + 1
                        .ENDR
                        .ENDC
                                                                PUSH    R0
                                                                MOV     U.UNUM(R5),HRQ.01
                                                                ADD     U.SUBU(R5),HRQ.01
                                                                MOV     #MESSAG,R0
                                                                CALL    HOSTRQ
                                                                POP     R0
                        .RADIX
                        .ENDM
;
;
```

```
                        ;ASSUME MACRO
                                .MACRO  ASSUME  P1,P2
                                .IF     NE,P1-P2
                                .ERROR  ;       THE ASSUMPTION THAT THE ABOVE VALUES ARE = IS FALSE
                                .ENDC
                                .ENDM
                        ;
                        ;       XOR THE CONTENTS OF TWO REGISTERS
                        ;
                                .MACRO  RXOR    REG1,REG2
                                MOV     REG2,-(SP)      ; SAVE REGISTER REG2
                                BIC     REG1,REG2               ; CLEAR COORESPONDING BITS IN REG2
                                BIC     (SP)+,REG1      ; CLEAR COORESPONDING BITS IN REG1
                                BIS     REG1,REG2               ; OR WHAT'S LEFT
                                .ENDM
                        ;
                        ;
                        ;       CONVERT .BLKW CALLS TO ACTUAL WORDS GENERATED
                        ;
                                .MACRO  .BLKW   COUNT
                                .NLIST
                                .REPT   COUNT
                                .WORD   0
                                .ENDR
                                .LIST
                                .ENDM
```

```
                                .SBTTL  START OF TEST CODE
                                ;THE FOLLOWING IS FOR DEBUG PURPOSES ONLY. CAN BE NOP OR BREAKPOINT.

000714  114007                        CLR     R0              ;CHANGE TO BREAKPOINT FOR DEBUG

                                ;INITIALIZE STACK

000715  104206  000745                MOV     #STACK,SP       ; SET UP STACK POINTER
000717  004530                        BR      START           ; BRANCH OVER SUPPORT CODE

                                ;STACK AREA

000720                                  .BLKW   21.                     ;STACK
000745  123456                  STACK:  .WORD   123456                  ;MARKER FOR STACK UNDERFLOW
```

```
                                        .SBTTL  RTDS  - REAL TIME DRIVE STATE ROUTINE WITH ERROR REPORTING (TEST 4)
000746  020775                  RTDS:   CALL    RTDSL           ; GET REAL TIME DRIVE STATE
000747  115002                          TST     R2              ; SEE IF ERROR OCCURRED
000750  010774                          BEQ     1$              ; IF NOT, BRANCH
000751  070756                          BMI     2$              ; IF RECEIVE ERROR, BRANCH
000752                                  CERROR  4,#ER17         ; REPORT NO CLOCKS ERROR MESSAGE
000752  104200  001607  001107                                                  MOV     #ER17,HRQ.04
000755  000761                          BR      3$              ; BRANCH
000756                          2$:     CERROR  4,#ER17A        ; REPORT RECEIVE ERRORS ERROR MESSAGE
000756  104200  001637  001107                                                  MOV     #ER17A,HRQ.04
000761  104200  044021  001105  3$:     MOV     #17.!FTLDEV+4000,HRQ.02 ; ERROR NUMBER SET
000764  104200  000764  001104          MOV     #.,HRQ.01       ; ADDRESS OF ERROR SET
000767  104200  060014  001103          MOV     #ERRMC,HRQ.RQ   ; DM REQUEST SET
000772                                  ENDERR  0
000772  114000  002230                                                  CLR     ERRPOS          ; CLEAR THE POSITION
000774  000000                  1$:     RETURN
                                        .SBTTL  RTDSL  - SAME AS RTDS BUT WITHOUT ERROR REPORTING
000775  104652  000025          RTDSL:  MOV     U.MASK(R5),R2   ; MOVE MASK TO R2
000777  021007                          CALL    RDSTAT          ; GET DRIVE STATUS
001000  103201  077674                  BIC     #^C<RCVRDY!ATTN!AVAIL!RWRDY>,R1 ; CLEAR UNUSED BITS
001002  115002                          TST     R2              ; TEST STATUS FOR ERROR
001003  011006                          BEQ     2$              ; IF NO ERROR, BRANCH
001004  104201  177777                  MOV     #-1,R1          ; FLAG AS INVALID
001006  000000                  2$:     RETURN
                                        .SBTTL  RDSTAT - GET DRIVE'S REAL TIME DRIVE STATE
001007                          RDSTAT:
                                        ;
                                        ;RETURN DRIVE STATUS
                                        ;STATUS RETURNED IN DM REGISTER 1
                                        ;
001007                                  PUSH    R0              ; SAVE R0
001007  100467                                                          MOV R0,-(SP)
001010  104207  024000                  MOV     #24000,R0       ; MOVE 1/2 SECOND COUNT TO R0
001012  060007                  1$:     XFC     STATUS          ; GET REAL TIME DRIVE STATE
001013  102201  000004                  BIT     #DCLOCK,R1      ; SEE IF DRIVE CLOCK IS PRESENT
001015  011027                          BEQ     3$              ; IF NOT, BRANCH
001016  102201  000400                  BIT     #RCVERR,R1      ; SEE IF RECEIVE ERROR
001020  011026                          BEQ     2$              ; IF NONE, BRANCH
001021  117407                          DEC     R0              ; DECREMENT COUNT
001022  051012                          BNE     1$              ; IF COUNT INCOMPLETE, BRANCH
001023  104202  177777                  MOV     #177777,R2      ; MARK AS RECEIVE ERROR
001025  001027                          BR      3$              ; BRANCH OVER 'NO ERRORS'
001026  114002                  2$:     CLR     R2              ; NO ERRORS
001027                          3$:     POP     R0              ; RESTORE R0
001027  104267                                                          MOV (SP)+,R0
001030  000000                          RETURN                  ; RETURN TO CALLING MODULE
```

```
      1                                 .SBTTL  HSTTLK - TALK TO THE HOST (INTERRUPT THE HOST BEFORE 3 MIN)
      2 001031                          TLKHST:
      3                                 ;
      4                                 ;       THIS ROUTINE INTERRUPTS THE HOST WITHOUT CONVEYING ANY USEFUL
      5                                 ;       INFORMATION - ALL IT DOES IS SAY 'I'M STILL ALIVE'
      6                                 ;
      7 001031                                  REPSFT  ,,,
        001031  114000  001105                                          CLR     HRQ.02
        001033  114000  001106                                          CLR     HRQ.03
        001035  114000  001107                                          CLR     HRQ.04
        001037  100467                                                          MOV R0,-(SF)
        001040  104657  000063                                          MOV     U.UNUM(R5),R0
        001042  105657  000050                                          ADD     U.SUBU(R5),R0
        001044  104070  001104                                          MOV     R0,HRQ.01
        001046  104207  060007                                          MOV     #T4SOFT,R0
        001050  021053                                                  CALL    HOSTRQ
        001051  104267                                                          MOV (SP)+,R0
      8 001052  000000                          RETURN
```

```
      1                                 .SBTTL  HOSTRQ - HOST REQUEST - REPORT ERRORS, MEGABYTES TRANSFERRED, ETC.
      2 001053                          HOSTRQ:
      3                                 ;
      4                                 ;SEND REQUEST BUFFER TO HOST AND WAIT FOR RESPONSE.
      5                                 ;CLEAR ARGUMENT AREA OF OUT BUFFER IN PREPARATION
      6                                 ;FOR NEXT HOSTRQ CALL.
      7                                 ;
      8                                 ;INPUTS:
      9                                 ;       R0 - HOST REQUEST NUMBER
     10                                 ;       OUT BUFFER LOADED WITH DATA
     11
     12 001053                                  PUSH    <R0,R1,R2>
        001053  100467                                                          MOV R0,-(SP)
        001054  100461                                                          MOV R1,-(SP)
        001055  100462                                                          MOV R2,-(SP)
     13 001056  104070  001103                  MOV     R0,HRQ.RQ               ; STORE REQUEST NUMBER IN BUFFER
     14 001060  104207  001103          SNDAGN: MOV     #HRQ.RQ,R0              ; SEND BUFFER TO HOST
     15 001062  104201  000043                  MOV     #BUFSIZ,R1              ; SIZE OF SEND BUFFER
     16 001064  060016                          XFC     MRD                     ; SEND BUFFER
     17 001065  115001                          TST     R1                      ; CHECK FOR ERROR
     18 001066  051060                          BNE     SNDAGN                  ; IF ERROR, TRY AGAIN
     19 001067  104207  001103                  MOV     #HRQ.RQ,R0              ; WAIT FOR RESPONSE FROM HOST
     20 001071  104201  000043                  MOV     #BUFSIZ,R1              ; SIZE OR RECEIVE BUFFER
     21 001073  060017                          XFC     MWR                     ; RECEIVE BUFFER
     22 001074  104200  177777  001103          MOV     #-1,HRQ.RQ              ; MAKE REQUEST ILLEGAL
     23 001077                                  POP     <R2,R1,R0>
        001077  104262                                                          MOV (SP)+,R2
        001100  104261                                                          MOV (SP)+,R1
        001101  104267                                                          MOV (SP)+,R0
     24 001102  000000                          RETURN
```

```
UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE 21
HOSTRQ - HOST REQUEST - REPORT ERRORS, MEGABYTES TRANSFERRED,

                                ;STORAGE AREA FOR MAINTENANCE WRITE AND READ BUFFERS
                                ;
                                ;HRQ BUFFER - DATA TO SEND/RECEIVE TO/FROM HOST
                                ;
001103  000000                  HRQ.RQ: .WORD 0                 ;HOST REQUEST CODE
001104  000000                  HRQ.01: .WORD 0                 ;DATA ARGUMENT  1
001105  000000                  HRQ.02: .WORD 0                 ;DATA ARGUMENT  2
001106  000000                  HRQ.03: .WORD 0                 ;DATA ARGUMENT  3
001107  000000                  HRQ.04: .WORD 0                 ;DATA ARGUMENT  4
001110  000000                  HRQ.05: .WORD 0                 ;DATA ARGUMENT  5
001111  000000                  HRQ.06: .WORD 0                 ;DATA ARGUMENT  6
001112  000000                  HRQ.07: .WORD 0                 ;DATA ARGUMENT  7
001113  000000                  HRQ.08: .WORD 0                 ;DATA ARGUMENT  8
001114  000000                  HRQ.09: .WORD 0                 ;DATA ARGUMENT  9
001115  000000                  HRQ.10: .WORD 0                 ;DATA ARGUMENT 10
001116  000000                  HRQ.11: .WORD 0                 ;DATA ARGUMENT 11
001117  000000                  HRQ.12: .WORD 0                 ;DATA ARGUMENT 12
001120  000000                  HRQ.13: .WORD 0                 ;DATA ARGUMENT 13
001121  000000                  HRQ.14: .WORD 0                 ;DATA ARGUMENT 14
001122  000000                  HRQ.15: .WORD 0                 ;DATA ARGUMENT 15
001123  000000                  HRQ.16: .WORD 0                 ;DATA ARGUMENT 16
001124  000000                  HRQ.17: .WORD 0                 ;DATA ARGUMENT 17
001125  000000                  HRQ.18: .WORD 0                 ;DATA ARGUMENT 18
001126  000000                  HRQ.19: .WORD 0                 ;DATA ARGUMENT 19
001127  000000                  HRQ.20: .WORD 0                 ;DATA ARGUMENT 20
001130  000000                  HRQ.21: .WORD 0                 ;DATA ARGUMENT 21
001131  000000                  HRQ.22: .WORD 0                 ;DATA ARGUMENT 22
001132  000000                  HRQ.23: .WORD 0                 ;DATA ARGUMENT 23
001133  000000                  HRQ.24: .WORD 0                 ;DATA ARGUMENT 24
001134  000000                  HRQ.25: .WORD 0                 ;DATA ARGUMENT 25
001135  000000                  HRQ.26: .WORD 0                 ;DATA ARGUMENT 26
001136  000000                  HRQ.27: .WORD 0                 ;DATA ARGUMENT 27
001137  000000                  HRQ.28: .WORD 0                 ;DATA ARGUMENT 28
001140  000000                  HRQ.29: .WORD 0                 ;DATA ARGUMENT 29
001141  000000                  HRQ.30: .WORD 0                 ;DATA ARGUMENT 30
001142  000000                  HRQ.31: .WORD 0                 ;DATA ARGUMENT 31
001143  000000                  HRQ.32: .WORD 0                 ;DATA ARGUMENT 32
001144  000000                  HRQ.33: .WORD 0                 ;DATA ARGUMENT 33
001145  000000                  HRQ.34: .WORD 0                 ;DATA ARGUMENT 34

        000043                  BUFSIZ  =       . - HRQ.RQ      ;SIZE OF BUFFER
```

```
                                .SBTTL  CMPEDC - EDC CALCULATION ROUTINE
001146                          CMPEDC:
                                ;
                                ;       THIS WILL COMPUTE THE EDC OF A SECTOR POINTED TO BY R0, RETURNING
                                ;       THE VALUE IN R2
                                ;
001146                                  PUSH    <R4,R1,R0>      ; SAVE R1 AND R0 AND R4
001146  100464                                                                  MOV R4,-(SP)
001147  100461                                                                  MOV R1,-(SP)
001150  100467                                                                  MOV R0,-(SP)
001151  104201  000400                  MOV     #256.,R1        ; COMPUTE OVER 256 WORDS
001153  104202  000105                  MOV     #INTEDC,R2      ; MOVE INITIAL EDC VALUE TO R2
001155  104274                  EDCLOP: MOV     (R0)+,R4        ; GET A WORD
001156                                  RXOR    R4,R2           ; EXCLUSIVE OR THE WORD WITH THE CURRENT EDC VALUE
001156  100462                          MOV     R2,-(SP)        ; SAVE REGISTER R2
001157  103042                          BIC     R4,R2           ; CLEAR COORESPONDING BITS IN R2
001160  103264                          BIC     (SP)+,R4        ; CLEAR COORESPONDING BITS IN R4
001161  101042                          BIS     R4,R2           ; OR WHAT'S LEFT
001162  105022                          ADD     R2,R2           ; SHIFT R2 LEFT BY 1
001163  041165                          BCC     ZEREDC          ; IF THE HIGH BIT WAS CLEAR, BRANCH
001164  115402                          INC     R2              ; SET LO BIT TO 1 (OLD HI BIT)
001165  117401                  ZEREDC: DEC     R1              ; DECREMENT COUNT
001166  051155                          BNE     EDCLOP          ; IF COUNT UNEXPIRED, BRANCH
001167                                  POP     <R0,R1,R4>      ; RESTORE R1 AND R0 AND R4
001167  104267                                                                  MOV (SP)+,R0
001170  104261                                                                  MOV (SP)+,R1
001171  104264                                                                  MOV (SP)+,R4
001172  000000                          RETURN
```

```
                                .SBTTL  TALK  - SDI LEVEL 2 INTERCHANGE ROUTINE
001173                          TALK:
                                ;
                                ;       TALK SENDS THE COMMAND TO THE DRIVE.  IF AN ERROR OCCURRS, R2 IS
                                ;       RETURNED NONZERO
                                ;
001173                                  PUSH    <R4,R5>         ; SAVE POINTER TO UNIT AND SUBUNIT PARAMETERS
001173  100464                                                                  MOV R4,-(SP)
001174  100465                                                                  MOV R5,-(SP)
001175  104652  000025                 MOV     U.MASK(R5),R2   ; GET UNIT SDI SELECT MASK
001177  104205  000001                 MOV     #1,R5           ; SEND TIMEOUT DEFAULT IS ONE
001201  106203  001616                 CMP     #SNDONE,R3      ; SEE IF ONLY TO BE SENT ONCE
001203  071206                         BMI     1$              ; IF SO, BRANCH
001204  104205  001750                 MOV     #MAXSND,R5      ; SEND MAXIMUM NUMBER OF TIMES (ONLINE)
001206  104137                 1$:     MOV     (R3),R0         ; POINTS TO SDI COMMAND BUFFER
001207                                  ASSUME  L2.OPC,0
001207  104631  000001                 MOV     L2.SLN(R3),R1   ; LOAD BYTE COUNT
001211  060004                         XFC     SEND            ; SEND SDI COMMAND
001212  115001                         TST     R1              ; SEE IF SDI COMMAND SENT SUCESSFULLY
001213  011237                         BEQ     2$              ; IF SO, BRANCH
001214  117405                         DEC     R5              ; DECREMENT TIMEOUT                     -
001215  051206                         BNE     1$              ; IF UNEXPIRED, BRANCH
001216                                  POP     R5              ; RESTORE R5
001216  104265                                                                  MOV (SP)+,R5
001217                                  PUSH    R3              ; SAVE SDI PACKET POINTER
001217  100463                                                                  MOV R3,-(SP)
001220  024243                         CALL    GOINIT          ; INIT THE DRIVE
001221                                  SOFTER  70              ; REPORT
001221  104200  004264  001107                                                  MOV     #ER70,HRQ.04
001224  104202  147746                                                          MOV     #70!ERSOFT+4000.,R2
001226  104020  001105                                                          MOV     R2,HRQ.02
001230  104200  001230  001104                                                  MOV     #.,HRQ.01
001233  104200  060013  001103                                                  MOV     #ERRMES,HRQ.RQ
001236  001440                         BR      11$             ; BRANCH
001237                          2$:     POP     R5              ; RESTORE R5
001237  104265                                                                  MOV (SP)+,R5
001240  104654  000033                 MOV     U.SDIS(R5),R4   ; R4 HAS SHORT TIMEOUT
001242  106203  001654                 CMP     #LONG,R3        ; SEE IF LONG TIMEOUT TO BE USED
001244  031247                         BPL     4$              ; IF SO, BRANCH
001245  104654  000034                 MOV     U.SDIL(R5),R4   ; R4 HAS LONG TIMEOUT
001247                          4$:     PUSH    R3              ; SAVE POINTER
001247  100463                                                                  MOV R3,-(SP)
001250  020746                         CALL    RTDS            ; GET REAL TIME DRIVE STATUS
001251  115002                         TST     R2              ; OK?
001252  011255                         BEQ     5$              ; IF SO, BRANCH
001253  024221                         CALL    GORTRY          ; ELSE, RETRY
001254  001610                         BR      17$             ; AND BRANCH DUE TO ERROR
001255  110601                 5$:     ROR     R1              ; IS RCVRDY SET?
001256                                  ASSUME RCVR!/,1
001256  041275                         BCC     6$              ; IF NOT, BRANCH
001257                                  SOFTER  77
001257  104200  004526  001107                                                  MOV     #ER77,HRQ.04
001262  104202  147755                                                          MOV     #77!ERSOFT+4000.,R2
001264  104020  001105                                                          MOV     R2,HRQ.02
001266  104200  001266  001104                                                  MOV     #.,HRQ.01
001271  104200  060013  001103                                                  MOV     #ERRMES,HRQ.RQ
001274  001440                         BR      11$             ; END ERROR
```

```
     42 001275  104207  001702          6$:     MOV     #ST,R0          ; POINT TO RECEIVE BUFFER
     43 001277  104631  000002                  MOV     L2.RLN(R3),R1   ; NUMBER OF WORDS IN RESPONSE
     44 001301  104652  000025                  MOV     U.MASK(R5),R2   ; STORE MASK IN R2
     45 001303  060005                          XFC     RCV             ; RECEIVE SDI RESPONSE
     46 001304  115001                          TST     R1              ; SEE IF SDI RESPONSE RECEIVED SUCESSFULLY
     47 001305  011475                          BEQ     13$             ; IF SO, BRANCH
     48 001306  106201  000001                  CMP     #1,R1           ; SEE IF TIMEOUT
     49 001310  051335                          BNE     7$              ; IF NOT, BRANCH
     50 001311  021031                          CALL    TLKHST          ; SAY 'I'M ALIVE' TO THE HOST
     51 001312                                  POP     R3              ; RESTORE R3
        001312  104263                                                                  MOV (SP)+,R3
     52 001313  117404                          DEC     R4              ; DECREMENT TIMEOUT VALUE
     53 001314  051247                          BNE     4$              ; IF TIMEOUT UNEXPIRED, BRANCH
     54 001315                                  PUSH    R3              ; SAVE R3
        001315  100463                                                                  MOV R3,-(SP)
     55 001316  024243                          CALL    GOINIT          ; INIT THE DRIVE
     56 001317                                  SOFTER  71              ; FLAG AS ERROR
        001317  104200  004300  001107                                          MOV     #ER71,HRQ.04
        001322  104202  147747                                                  MOV     #71!ERSOFT+4000.,R2
        001324  104020  001105                                                  MOV     R2,HRQ.02
        001326  104200  001326  001104                                          MOV     #.,HRQ.01
        001331  104200  060013  001103                                          MOV     #ERRMES,HRQ.RQ
     57 001334  001440                          BR      11$             ; BRANCH
     58 001335  106201  000002          7$:     CMP     #2,R1           ; SEE IF FIRST WORD NOT START FRAME
     59 001337  051356                          BNE     8$              ; IF NOT, BRANCH
     60 001340                                  SOFTER  72              ; REPORT FIRST WORD NOT START FRAME
        001340  104200  004315  001107                                          MOV     #ER72,HRQ.04
        001343  104202  147750                                                  MOV     #72!ERSOFT+4000.,R2
        001345  104020  001105                                                  MOV     R2,HRQ.02
        001347  104200  001347  001104                                          MOV     #.,HRQ.01
        001352  104200  060013  001103                                          MOV     #ERRMES,HRQ.RQ
     61 001355  001440                          BR      11$             ; BRANCH
     62 001356  106201  000004          8$:     CMP     #4,R1           ; SEE IF FRAMING ERROR
     63 001360  051377                          BNE     9$              ; IF NOT, BRANCH
     64 001361                                  SOFTER  73              ; REPORT FRAMING ERROR
        001361  104200  004345  001107                                          MOV     #ER73,HRQ.04
        001364  104202  147751                                                  MOV     #73!ERSOFT+4000.,R2
        001366  104020  001105                                                  MOV     R2,HRQ.02
        001370  104200  001370  001104                                          MOV     #.,HRQ.01
        001373  104200  060013  001103                                          MOV     #ERRMES,HRQ.RQ
     65 001376  001440                          BR      11$             ; BRANCH
     66 001377  106201  000010          9$:     CMP     #10,R1          ; SEE IF CHECKSUM ERROR
     67 001401  051420                          BNE     10$             ; IF NOT, BRANCH
     68 001402                                  SOFTER  74              ; REPORT CHECKSUM ERROR
        001402  104200  004371  001107                                          MOV     #ER74,HRQ.04
        001405  104202  147752                                                  MOV     #74!ERSOFT+4000.,R2
        001407  104020  001105                                                  MOV     R2,HRQ.02
        001411  104200  001411  001104                                          MOV     #.,HRQ.01
        001414  104200  060013  001103                                          MOV     #ERRMES,HRQ.RQ
     69 001417  001440                          BR      11$             ; BRANCH
     70 001420  106201  000020          10$:    CMP     #20,R1          ; SEE IF BUFFER TOO SMALL
     71 001422  051447                          BNE     12$             ; IF NOT, BRANCH
     72 001423                                  SOFTER  75              ; REPORT BUFFER TO SMALL
        001423  104200  004416  001107                                          MOV     #ER75,HRQ.04
        001426  104202  147753                                                  MOV     #75!ERSOFT+4000.,R2
        001430  104020  001105                                                  MOV     R2,HRQ.02
        001432  104200  001432  001104                                          MOV     #.,HRQ.01
```

```
      001435  104200  060013  001103                                                    MOV     #ERRMES,HRQ.RQ
   73 001440                                11$:    ENDERR  6               ; FLAG END OF REPORTING BUFFER
      001440  104200  000051  001111                                                    MOV     #SER22,HRQ.06
      001443  104200  000007  002230                                                    MOV     #6+1,ERRPOS     ; SET THE POSITION
   74 001446  001537                               BR      14$             ; EXIT
   75 001447                                12$:    HARDER  78              ; UNKNOWN ERROR RETURNED BY UDA
      001447  104200  004562  001107                                                    MOV     #ER78,HRQ.04
      001452  104202  107756                                                            MOV     #78!ERHARD+4000.,R2
      001454  104020  001105                                                            MOV     R2,HRQ.02
      001456  104200  001456  001104                                                    MOV     #.,HRQ.01
      001461  104200  060014  001103                                                    MOV     #ERRMC,HRQ.RQ
   76 001464                                       CERROR  6,R1            ; REPORT ERROR
      001464  104010  001111                                                            MOV     R1,HRQ.06
   77 001466                                       ENDERR  7               ; FLAG END OF REPORTING BUFFER
      001466  104200  000051  001112                                                    MOV     #SER22,HRQ.07
      001471  104200  000010  002230                                                    MOV     #7+1,ERRPOS     ; SET THE POSITION
   78 001474  001537                               BR      14$             ; EXIT
   79 001475  114002                           13$:    CLR     R2              ; FLAG AS NO ERROR OCCURED
   80 001476  106637  000003                           CMP     L2.ERS(R3),R0   ; SEE IF COMMAND ACCEPTED
   81 001500  011557                               BEQ     16$             ; IF SO, BRANCH
   82 001501                                       SOFTER  76              ; REPORT
      001501  104200  004447  001107                                                    MOV     #ER76,HRQ.04
      001504  104202  147754                                                            MOV     #76!ERSOFT+4000.,R2
      001506  104020  001105                                                            MOV     R2,HRQ.02
      001510  104200  001510  001104                                                    MOV     #.,HRQ.01
      001513  104200  060013  001103                                                    MOV     #ERRMES,HRQ.RQ
   83 001516                                       CERROR  6,<L2.ERS(R3),R0,#SER22> ; REPORT FURTHER ERRORS
      001516  104630  000003  001111                                                    MOV     L2.ERS(R3),HRQ.06
      001521  104070  001112                                                            MOV     R0,HRQ.07
      001523  104200  000051  001113                                                    MOV     #SER22,HRQ.08
   84 001526  104200  100011  002230               MOV     #<VALID+11>,ERRPOS ; FLAG AS STATUS GOOD
   85 001531  106207  000175                       CMP     #UNSSUC,R0      ; SEE IF UNSUCCESSFUL RESPONSE
   86 001533  011537                               BEQ     14$             ; IF SO, BRANCH
   87 001534  103200  100000  002230               BIC     #VALID,ERRPOS   ; GET STATUS FOR PRINTING
   88 001537  024221                           14$:    CALL    GORTRY          ; RETRY THIS COMMAND (MINIMUM)
   89 001540                                       POP     R3              ; RESTORE POINTER TO COMMAND
      001540  104263                                                                   MOV (SP)+,R3
   90 001541  104633  000004                       MOV     L2.EOF(R3),R3   ; GET SDI ERROR OFFSET
   91 001543  105053                               ADD     R5,R3           ; POINT TO ERROR WORD
   92 001544  104134                               MOV     (R3),R4         ; GET ERROR COUNT
   93 001545  115404                               INC     R4              ; INCREMENT COUNT
   94 001546  106204  000002                       CMP     #2,R4           ; SEE IF MAX
   95 001550  041555                               BCC     15$             ; IF NOT, BRANCH
   96 001551  103200  100000  001105               BIC     #C2DFTL,HRQ.02  ; CHANGE ERROR TO A DEVICE FATAL
   97 001554  114004                               CLR     R4              ; IN CASE DROPS DISABLED, CLEAR ERROR COUNT
   98 001555  100134                           15$:    MOV     R4,(R3)         ; SAVE COUNT
   99 001556  001610                               BR      17$             ; EXIT
  100 001557                                   16$:    POP     R3              ; RESTORE POINTER TO COMMAND
      001557  104263                                                                   MOV (SP)+,R3
  101 001560  104633  000004                       MOV     L2.EOF(R3),R3   ; GET SDI ERROR OFFSET
  102 001562  105053                               ADD     R5,R3           ; POINT TO ERROR WORD
  103 001563  104134                               MOV     (R3),R4         ; GET ERROR COUNT
  104 001564  011610                               BEQ     17$             ; IF NO RETRIES, BRANCH
  105 001565  100132                               MOV     R2,(R3)         ; ZERO RETRIES
  106 001566                                       REPSFT  SOFT,,,COMM     ; REPORT SOFT AND COMMUNICATION ERRORS
      001566  104200  000001  001105                                                    MOV     #1,HRQ.02
      001571  114000  001106                                                            CLR     HRQ.03
```

```
        001573  114000  001107                                                 CLR     HRQ.04
        001575  100467                                                                 MOV R0,-(SP)
        001576  104657  000063                                                 MOV     U.UNUM(R5),R0
        001600  105657  000050                                                 ADD     U.SUBU(R5),R0
        001602  104070  001104                                                 MOV     R0,HRQ.01
        001604  104207  060007                                                 MOV     #T4SOFT,R0
        001606  021053                                                         CALL    HOSTRQ
        001607  104267                                                                 MOV (SP)+,R0
    107 001610                          17$:    POP     R4              ; RESTORE R4
        001610  104264                                                                 MOV (SP)+,R4
    108 001611  000000                          RETURN
```

```
                                .SBTTL  SDI PROTOCOL MESSAGE TABLES AND RESPONSE BUFFER
                                ;
                                ;       MESSAGE TABLES
                                ;
        000000                  SDI$    =       0               ; INITIAL OFFSET
001612                          CR.ONL: MSG ONL,2,7,COMPLT      ; BRING DRIVE ONLINE
001612  001700                          .WORD   ONL                     ; ADDRESS OF COMMAND
001613  000002                          .WORD   2                       ; SIZE OF COMMAND IN BYTES
001614  000007                          .WORD   7                       ; SIZE OF REPLY IN WORDS
001615  000176                          .WORD   COMPLT                  ; SUCCESSFUL COMPLETION CODE
001616  000035                          .WORD   SDI$+U.SDI2             ; RETRY COUNT OFFSET
        001616                  SNDONE  =       .-1     ; PREVIOUS COMMANDS CAN BE SENT TO AN OFFLINE DR...
001617                          CR.GST: MSG GST,1,7,STSRES      ; GET STATUS
001617  001670                          .WORD   GST                     ; ADDRESS OF COMMAND
001620  000001                          .WORD   1                       ; SIZE OF COMMAND IN BYTES
001621  000007                          .WORD   7                       ; SIZE OF REPLY IN WORDS
001622  000366                          .WORD   STSRES                  ; SUCCESSFUL COMPLETION CODE
001623  000036                          .WORD   SDI$+U.SDI2             ; RETRY COUNT OFFSET
001624                          CR.CLR: MSG DRC,2,7,COMPLT      ; DRIVE CLEAR
001624  001666                          .WORD   DRC                     ; ADDRESS OF COMMAND
001625  000002                          .WORD   2                       ; SIZE OF COMMAND IN BYTES
001626  000007                          .WORD   7                       ; SIZE OF REPLY IN WORDS
001627  000176                          .WORD   COMPLT                  ; SUCCESSFUL COMPLETION CODE
001630  000037                          .WORD   SDI$+U.SDI2             ; RETRY COUNT OFFSET
001631                          CR.MOD: MSG MOD,3,7,COMPLT      ; CHANGE MODE
001631  001664                          .WORD   MOD                     ; ADDRESS OF COMMAND
001632  000003                          .WORD   3                       ; SIZE OF COMMAND IN BYTES
001633  000007                          .WORD   7                       ; SIZE OF REPLY IN WORDS
001634  000176                          .WORD   COMPLT                  ; SUCCESSFUL COMPLETION CODE
001635  000040                          .WORD   SDI$+U.SDI2             ; RETRY COUNT OFFSET
001636                          CR.SEK: MSG INS,6,7,COMPLT      ; INITIATE SEEK
001636  001672                          .WORD   INS                     ; ADDRESS OF COMMAND
001637  000006                          .WORD   6                       ; SIZE OF COMMAND IN BYTES
001640  000007                          .WORD   7                       ; SIZE OF REPLY IN WORDS
001641  000176                          .WORD   COMPLT                  ; SUCCESSFUL COMPLETION CODE
001642  000041                          .WORD   SDI$+U.SDI2             ; RETRY COUNT OFFSET
001643                          CR.ERR: MSG ERR,2,7,COMPLT      ; ERROR RECOVERY
001643  001676                          .WORD   ERR                     ; ADDRESS OF COMMAND
001644  000002                          .WORD   2                       ; SIZE OF COMMAND IN BYTES
001645  000007                          .WORD   7                       ; SIZE OF REPLY IN WORDS
001646  000176                          .WORD   COMPLT                  ; SUCCESSFUL COMPLETION CODE
001647  000042                          .WORD   SDI$+U.SDI2             ; RETRY COUNT OFFSET
001650                          CR.DIS: MSG DIS,2,7,COMPLT      ; DISCONNECT DRIVE
001650  001662                          .WORD   DIS                     ; ADDRESS OF COMMAND
001651  000002                          .WORD   2                       ; SIZE OF COMMAND IN BYTES
001652  000007                          .WORD   7                       ; SIZE OF REPLY IN WORDS
001653  000176                          .WORD   COMPLT                  ; SUCCESSFUL COMPLETION CODE
001654  000043                          .WORD   SDI$+U.SDI2             ; RETRY COUNT OFFSET
        001654                  LONG    =       .-1     ; ALL COMMANDS BEYOND THIS POINT ARE LONG TIMEOUT
001655                          CR.INR: MSG INR,1,7,COMPLT      ; INITIATE RECALIBRATE
001655  001671                          .WORD   INR                     ; ADDRESS OF COMMAND
001656  000001                          .WORD   1                       ; SIZE OF COMMAND IN BYTES
001657  000007                          .WORD   7                       ; SIZE OF REPLY IN WORDS
001660  000176                          .WORD   COMPLT                  ; SUCCESSFUL COMPLETION CODE
001661  000044                          .WORD   SDI$+U.SDI2             ; RETRY COUNT OFFSET
        000010                  NUML2S  =       SDI$            ; SAVE NUMBER OF LEVEL 2 SDI COMMANDS
                                ;
```

```
                                        ;       LEVEL 2 COMMAND MESSAGE DATA STRUCTURES
                                        ;
001662    000       204         DIS:    .BYTE   0,DISCON        ; DISCONNECT DRIVE
001663  000000                          .WORD   0               ; DO NOT SPIN DOWN DRIVE
001664    000       201         MOD:    .BYTE   0,CHGMOD        ; CHANGE MODE
001665  000000                  WRITBT: .WORD   0               ; MODE
001666    000       005         DRC:    .BYTE   0,DRVCLR        ; DRIVE CLEAR
001667  000000                  DCLR:   .WORD   0
001670    000       011         GST:    .BYTE   0,GETSTA        ; GET STATUS
001671    000       216         INR:    .BYTE   0,IRECLB        ; INITIATE RECALIBRATE
001672    000       012         INS:    .BYTE   0,INSEEK        ; INITIATE SEEK
001673  000000                  LOCYL:  .WORD   0               ; INS CYLINDER/HEAD ARGUMENTS
001674  000000                          .WORD   0               ; HI CYLINDER
001675  000000                          .WORD   0               ; GROUP
001676    000       006         ERR:    .BYTE   0,ERECOV        ; ERROR RECOVERY
001677  000000                  ERRLEV: .WORD   0
001700    000       213         ONL:    .BYTE   0,DRVONL        ; ONLINE COMMAND
001701  000012                          .WORD   10.             ; TIMEOUT VALUE


001702                          ST:     .BLKW   19.             ; RESPONSE BUFFER
```

```
                                .SBTTL  BLKCHK - SEE IF A BLOCK WITH ERROR IS KNOWN TO BE BAD
001725                          BLKCHK:
                                ;
                                ;       BLKCHK CHECKS THE BLOCK IN U.LBN TO ASSURE THAT IT IS NOT A BAD
                                ;       BLOCK.  IF SO, R2 IS RETURNED NONZERO
                                ;
001725                                  PUSH    <R2,R1,R0>      ; SAVE ALL REG'S
001725  100462                                                                  MOV R2,-(SP)
001726  100461                                                                  MOV R1,-(SP)
001727  100467                                                                  MOV R0,-(SP)
001730  104651  000046                    MOV     U.PARM(R5),R1   ; GET UNIT PARAMETERS
001732  102201  000200                    BIT     #RBNBN,R1       ; SEE IF ON RBN
001734  051754                          BNE     1$              ; IF SO, UNKNOWN BAD BLOCK
001735  104641  000012                    MOV     S.BADP(R4),R1   ; GET BAD BLOCK POINTER
001737  011754                          BEQ     1$              ; IF NO BAD BLOCKS, BRANCH
001740  104202  000002                    MOV     #RW.LOW,R2      ; POINT TO LBN TO BE TESTED
001742  105072                          ADD     R0,R2           ; POINT TO LBN TO BE TESTED
001743  021765                  3$:     CALL    CMP2            ; COMPARE BAD BLOCK TO LBN
001744  011757                          BEQ     2$              ; IF EQUAL, BRANCH
001745                                  BCS     1$              ; IF LIST GREATER THAN BLOCK, IT'S OK
001745  041747                                                                  BCC     .+2
001746  001754                                                                  BR      1$
001747  105201  000002                    ADD     #2,R1           ; POINT TO NEXT BAD BLOCK
001751  104617  177777                    MOV     -1(R1),R0       ; SEE IF EOL
001753  031743                          BPL     3$              ; IF NOT, BRANCH
001754  104207  000001          1$:     MOV     #1,R0           ; SET UP FOR CARRY TO BE SET (UNKNOWN BAD BLOCK)
001756  001760                          BR      4$
001757  114007                  2$:     CLR     R0              ; SET UP FOR THE CARRY TO BE CLEAR (BAD BLOCK KNOWN)
001760  110607                  4$:     ROR     R0              ; SET CARRY TO REFLECT KNOWLEDGE OF BAD BLOCK
001761                                  POP     <R0,R1,R2>      ; RESTORE
001761  104267                                                                  MOV (SP)+,R0
001762  104261                                                                  MOV (SP)+,R1
001763  104262                                                                  MOV (SP)+,R2
001764  000000                          RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  CMP2   - 28 BIT COMPARE
001765                                  CMP2:
                                        ;
                                        ;       CMP2 COMPARES A 28 BIT NUMBER POINTED TO BY R2 TO A 28 BIT NUMBER
                                        ;       POINTED TO BY R1.  BOTH NUMBERS ARE LOW ORDER WORD FOLLOWED BY HIGH
                                        ;       WORD (POINTER TO LOW WORD) AND THE HIGH 4 BITS <31:28> OF THE
                                        ;       WORD POINTED TO BY R1 ARE STRIPPED OFF (THIS IS TO ELIMINATE THE
                                        ;       END-OF-LIST FLAG ON THE B/E SETS AND BAD BLOCKS)
                                        ;
001765  104617  000001                        MOV     1(R1),R0        ; MOVE HIGH ORDER WORD TO R0
001767  103207  170000                        BIC     #^CHBHINB,R0    ; CLEAR UNUSED BITS
001771  106627  000001                        CMP     1(R2),R0        ; COMPARE HIGH ORDER WORRD TO R0
001773  051776                              BNE     1$              ; IF UNEQUAL, BRANCH
001774  104117                              MOV     (R1),R0         ; MOVE LOW ORDER WORD TO R0
001775  106127                              CMP     (R2),R0         ; COMPARE LOW ORDER WORD TO R0
001776  000000                  1$:         RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  BULDUM - BUILD THE DUMMY SDI CONTROL BLOCK FOR READS AND WRITES
001777                                  BULDUM:
                                        ;
                                        ;       BULDUM WILL BUILD THE DUMMY SDI CONTROL BLOCK
                                        ;
001777  114002                                  CLR     R2                      ; NO RBN'S
002000  104643  000000                          MOV     S.PARM(R4),R3           ; GET SUBUNIT PARAMETERS
002002  102203  020000                          BIT     #DCYLS,R3               ; SEE IF IN DIAGNOSTIC AREA
002004  052013                                  BNE     1$                      ; IF SO, BRANCH
002005  104642  000007                          MOV     S.SCHR(R4),R2           ; GET SUBUNIT CHARACTERISTICS
002007  104622  000004                          MOV     RBNTRK(R2),R2           ; GET RBN'S/TRACK
002011  103202  177600                          BIC     #HIBYTE!200,R2          ; CLEAR UNUSED BITS
002013  105642  000006              1$:         ADD     S.TRKL(R4),R2           ; GET SECTORS/TRACK
002015  105022                                  ADD     R2,R2                   ; DOUBLE SECTORS/TRACK FOR HEADER COMPARE LIMIT
002016  104020  002210                          MOV     R2,DUMSDI+D.LIMT                ; MOVE TO DUMMY SEARCH LIMIT
002020  104643  000007                          MOV     S.SCHR(R4),R3           ; R3 POINTS TO SUBUNIT CHARACTERISTICS
002022  107203  000005                          SUB     #5,R3                   ; R3 POINTS TO SUB CHAR - 5
002024  104030  002211                          MOV     R3,DUMSDI+D.SCHR                ; MOVE TO DUMMY SUB CHAR POINTER
002026  000000                                  RETURN                          ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  CALC   - CALCULATE THE CYL, GRP AND TRACK FOR THE GIVEN L/RBN
002027                                  CALC:
                                        ;
                                        ;       CALC CALCULATES THE CYLINDER, GROUP, TRACK AND SECTOR OF THE BLOCK
                                        ;
002027  115007                                  TST     R0              ; SEE IF CALCULATION AREA MUST BE SET UP
002030  052164                                  BNE     MOVOUT          ; IF NOT, BRANCH
002031  104070  002200                          MOV     R0,RBNFLG       ; SET RBN FLAG AS ZERO
002033  104643  000007                          MOV     S.SCHR(R4),R3   ; R3 POINTS TO SUBUNIT'S CHARACTERISTICS
002035  104637  000004                          MOV     RBNTRK(R3),R0   ; GET RBN'S PER TRACK
002037  103207  177600                          BIC     #HIBYTE!200,R0  ; CLEAR UNUSED BITS
002041  105637  000011                          ADD     LBNTRK(R3),R0   ; ADD LBN'S PER TRACK
002043  103207  177400                          BIC     #HIBYTE,R0      ; CLEAR UNUSED BITS
002045  104070  002217                          MOV     R0,SCR1         ; SAVE
002047  104147                                  MOV     (R4),R0         ; GET SUBUNIT PARAMETERS
002050                                          ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
002050  102207  020000                          BIT     #DCYLS,R0       ; SEE IF USING THE DIAGNOSTIC CYLINDERS
002052  052123                                  BNE     MOVDBN          ; IF SO, BRANCH
002053  104637  000011                          MOV     LBNTRK(R3),R0   ; MOVE LBN'S PER TRACK TO R0
002055  103207  177400                          BIC     #HIBYTE,R0      ; CLEAR UNUSED BITS
002057  104651  000046                          MOV     U.PARM(R5),R1   ; MOVE UNIT PARAMETERS TO R0
002061  102201  000200                          BIT     #RBNBN,R1       ; SEE IF BLOCK REVECTORED
002063  012101                                  BEQ     LNCYL           ; IF NOT, BRANCH
002064  104070  002200                          MOV     R0,RBNFLG       ; STORE LBN'S PER TRACK IN RBN FLAG AREA
002066  104637  000004                          MOV     RBNTRK(R3),R0   ; MOVE RBN'S PER TRACK TO R0
002070  103207  177600                          BIC     #177600,R0      ; CLEAR THE HIGHER BITS, LEAVING RBN'S/TRACK
002072  104070  002177                          MOV     R0,LRDTRK       ; MOVE RBN'S PER TRACK TO CALCULATION AREA
002074  104637  000003                          MOV     HIRBN(R3),R0    ; GET HI RBN
002076  103207  170377                          BIC     #HBLONB,R0      ; CLEAR UNUSED BITS
002100  002107                                  BR      LNCONT          ; BRANCH
002101  104070  002177          LNCYL:  MOV     R0,LRDTRK       ; MOVE LBN'S PER TRACK TO CALCULATION AREA
002103  104637  000002                          MOV     HILBN(R3),R0    ; GET HI LBN
002105  103207  170377                          BIC     #HBLONB,R0      ; CLEAR UNUSED BITS
002107  104070  002172          LNCONT: MOV     R0,HIBN         ; SAVE
002111  114007                                  CLR     R0              ; LOW ORDER CYLINDER IS ZERO
002112  104070  002173                          MOV     R0,STACYL       ; SAVE LO ORDER STARTING CYLINDER
002114  104637  000001                          MOV     HICYL(R3),R0    ; R0 CONTAINS HI CYLINDER BITS
002116  103207  007777                          BIC     #HBHINB,R0      ; CLEAR UNUSED BITS
002120  104070  002174                          MOV     R0,STACYL+1     ; MOVE HI STARTING CYLINDER TO UNIT PARAMETERS
002122  002164                                  BR      MOVOUT          ; BRANCH
002123  104637  000004          MOVDBN: MOV     RBNTRK(R3),R0   ; MOVE NUMBER OF RBN'S PER TRACK TO R0
002125  103207  177600                          BIC     #177600,R0      ; CLEAR THE HIGHER BITS, LEAVING RBN'S/TRACK
002127  104631  000011                          MOV     LBNTRK(R3),R1   ; ADD LBN'S/TRK, GIVING DBN'S/TRK
002131  103201  177400                          BIC     #HIBYTE,R1      ; CLEAR UNUSED BITS
002133  105017                                  ADD     R1,R0           ; ADD LBNS/TRK TO RBNS/TRK TO GIVE DBNS/TRK
002134  104070  002177                          MOV     R0,LRDTRK       ; SAVE DBNS/TRK IN COMPUTATIONAL AREA
002136  104647  000002                          MOV     S.SDCL(R4),R0   ; GET LO ORDER STARTING DIAGNOSTIC CYLINDER
002140  104070  002173                          MOV     R0,STACYL       ; SAVE LO ORDER CYL
002142  104637  000001                          MOV     HICYL(R3),R0    ; GET HI CYLINDER BITS
002144  103207  007777                          BIC     #HBHINB,R0      ; CLEAR UNUSED BITS
002146  101647  000003                          BIS     S.SDCL+1(R4),R0 ; GET HI ORDER STARTING DIAGNOSTIC CYLINDER
002150  104070  002174                          MOV     R0,STACYL+1     ; SAVE HI ORDER CYL
002152  104637  000003                          MOV     HIDBN(R3),R0    ; GET HI DBN BITS
002154  110607                                  ROR     R0              ; ROTATE TO CORRECT POSITION
002155  110607                                  ROR     R0              ; ROTATE TO CORRECT POSITION
002156  110607                                  ROR     R0              ; ROTATE TO CORRECT POSITION
002157  110607                                  ROR     R0              ; ROTATE TO CORRECT POSITION
```

```
     58 002160  103207  170377                    BIC     #HBLONB,R0      ; CLEAR UNUSED BITS
     59 002162  104070  002172                    MOV     R0,HIBN         ; SAVE
     60 002164  104207  002173          MOVOUT:   MOV     #CALCSC,R0      ; POINT TO CALCULATION AREA
     61 002166  104641  000007                    MOV     S.SCHR(R4),R1   ; POINT TO SUBUNIT CHAR TABLE
     62 002170  060020                            XFC     CVT             ; CALCULATE
     63 002171  000000                            RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                .SBTTL  TEST 4 STORAGE AREA FOR VARIABLES
002172                  HIBN:   .BLKW   1               ; HI BN BITS <27:24>
002173                  CALCSC:                         ; CALCULATION AREA FOR CYL, GRP, TRK
002173                  STACYL: .BLKW   2               ; STARTING CYLINDER NUMBER
002175                  CURBN:  .BLKW   2               ; L/R/DBN
002177                  LRDTRK: .BLKW   1               ; SECTORS PER TRACK
002200                  RBNFLG: .BLKW   1               ; RBN FLAG
002201                  CYL:    .BLKW   2               ; CYLINDER
002203                  GROUP:  .BLKW   1               ; GROUP
002204                  TRACK:  .BLKW   1               ; TRACK
002205                  SECTOR: .BLKW   1               ; SECTOR
002206                  INDEX:  .BLKW   1               ; INDEX

002207                  DUMSDI: .BLKW   8.              ; DUMMY SDI AREA

002217                  SCR1:   .BLKW   1.              ; XFC READ AND WRITE WILL WRITE THE
002220                  SCR2:   .BLKW   1.              ; REVECTORED LBN IN THIS SPACE,
                                                        ; OTHERWISE, USE IT FOR SCRATCH

002221                  PNUM:   .BLKW   1               ; PATTERN NUMBER STORAGE AREA FOR ERRORS
002222                  LSTOVL: .BLKW   1               ; LAST OVERLAY CALLED NUMBER
002223                  M.PARM: .BLKW   1               ; MASTER PARAMETERS
002224  034245          LOSEED: .WORD   34245           ; LO ORDER RANDOM NUMBER SEED
002225  061453          HISEED: .WORD   61453           ; HI ORDER RANDOM NUMBER SEED
002226                  ERMODE: .BLKW   1               ; UNEXPECTED ATTENTION FLAG
002227  007774          MEMPOL: .WORD   HIMEM           ; MEMORY POOL FOR UNIT DATASTRUCTURES
002230                  ERRPOS: .BLKW   1               ; RTDS AND STATUS POSITION
002231                          ASSUME .-DUMSDI,18.
                        ; *** NEXT WORD REQUIRED BY UDA TO BE 18 PAST DUMMY SDI BLOCK START
002231  002203                  .WORD   DUMSDI-4        ; POINTER FOR LBN REVECTOR INFORMATION
                                ;       BELEIVE IT OR NOT, THIS WILL WRITE THE REVECTORED LBN
                                ;       IN LOCATIONS DUMSDI+8 AND DUMSDI+9
                                ;       EG. DUMSDI+18. POINTS TO 12 SHORT OF WHERE TO PUT THE
                                ;               REVECTOR INFORMATION


002232                  CHAINS: .BLKW   1               ; READ/WRITE CHAIN HEADER
002233                  SECMAX: .BLKW   1               ; MAXIMUM NUMBER OF BUFFERS THAT CAN BE READ
002234                  CHNMAX: .BLKW   1               ; MAXIMUM NUMBER OF BUFFERS THAT CAN BE WRITTEN
002235                  MAXSEC: .BLKW   1               ; MAXIMUM NUMBER OF SECTORS THIS PASS
```

```
                                .SBTTL  DATA PATTERNS THAT WILL BE WRITTEN TO THE DRIVE
002236  002256  PATPTR: .WORD   PAT0                    ; POINTERS TO DATA PATTERNS
002237  002300          .WORD   PAT1
002240  002303          .WORD   PAT2
002241  002306          .WORD   PAT3
002242  002311          .WORD   PAT4
002243  002333          .WORD   PAT5
002244  002355          .WORD   PAT6
002245  002377          .WORD   PAT7
002246  002402          .WORD   PAT8
002247  002424          .WORD   PAT9
002250  002427          .WORD   PAT10
002251  002451          .WORD   PAT11
002252  002454          .WORD   PAT12
002253  002476          .WORD   PAT13
002254  002520          .WORD   PAT14
002255  002525          .WORD   PAT15
                        ;
                        ;       DATA PATTERNS (EDC OF PATTERN, LENGTH OF PATTERN IN WORDS, FOLLOWED BY PATTERN)
                        ;
002256  177777  PAT0:    .WORD   177777          ; DUMMY EDC
002257  000001          .WORD   1               ; USER DEFINEABLE PATTERN
002260  000000          .WORD   0
002261  000000          .WORD   0
002262  000000          .WORD   0
002263  000000          .WORD   0
002264  000000          .WORD   0
002265  000000          .WORD   0
002266  000000          .WORD   0
002267  000000          .WORD   0
002270  000000          .WORD   0
002271  000000          .WORD   0
002272  000000          .WORD   0
002273  000000          .WORD   0
002274  000000          .WORD   0
002275  000000          .WORD   0
002276  000000          .WORD   0
002277  000000          .WORD   0
002300  115337  PAT1:    .WORD   115337          ; EDC FOR PATTERN 1
002301  000001          .WORD   1               ; B'1000101110001011'
002302  105613          .WORD   105613
002303  010524  PAT2:    .WORD   010524          ; EDC FOR PATTERN 2
002304  000001          .WORD   1
002305  031463          .WORD   031463          ; B'0011001100110011'
002306  001747  PAT3:    .WORD   001747          ; EDC FOR PATTERN 3
002307  000001          .WORD   1
002310  030221          .WORD   030221
002311  135776  PAT4:    .WORD   135776          ; EDC FOR PATTERN 4
002312  000020          .WORD   16.             ; SHIFTING ONES
002313  000001          .WORD   000001
002314  000003          .WORD   000003
002315  000007          .WORD   000007
002316  000017          .WORD   000017
002317  000037          .WORD   000037
002320  000077          .WORD   000077
002321  000177          .WORD   000177
002322  000377          .WORD   000377
```

DATA PATTERNS THAT WILL BE WRITTEN TO THE DRIVE

```
 58 002323  000777                          .WORD   000777
 59 002324  001777                          .WORD   001777
 60 002325  003777                          .WORD   003777
 61 002326  007777                          .WORD   007777
 62 002327  017777                          .WORD   017777
 63 002330  037777                          .WORD   037777
 64 002331  077777                          .WORD   077777
 65 002332  177777                          .WORD   177777
 66 002333  052420                PAT5:     .WORD   052420          ; EDC FOR PATTERN 5
 67 002334  000020                          .WORD   16.             ; SHIFTING ZEROS
 68 002335  177776                          .WORD   177776
 69 002336  177774                          .WORD   177774
 70 002337  177770                          .WORD   177770
 71 002340  177760                          .WORD   177760
 72 002341  177740                          .WORD   177740
 73 002342  177700                          .WORD   177700
 74 002343  177600                          .WORD   177600
 75 002344  177400                          .WORD   177400
 76 002345  177000                          .WORD   177000
 77 002346  176000                          .WORD   176000
 78 002347  174000                          .WORD   174000
 79 002350  170000                          .WORD   170000
 80 002351  160000                          .WORD   160000
 81 002352  140000                          .WORD   140000
 82 002353  100000                          .WORD   100000
 83 002354  000000                          .WORD   000000
 84 002355  114734                PAT6:     .WORD   114734          ; EDC FOR PATTERN 6
 85 002356  000020                          .WORD   16.             ; ALTERNATING ZERO WORD AND ONE WORD IN
 86 002357  000000                          .WORD   000000          ; 3-2-1-1-1
 87 002360  000000                          .WORD   000000
 88 002361  000000                          .WORD   000000
 89 002362  177777                          .WORD   177777
 90 002363  177777                          .WORD   177777
 91 002364  177777                          .WORD   177777
 92 002365  000000                          .WORD   000000
 93 002366  000000                          .WORD   000000
 94 002367  177777                          .WORD   177777
 95 002370  177777                          .WORD   177777
 96 002371  000000                          .WORD   000000
 97 002372  177777                          .WORD   177777
 98 002373  000000                          .WORD   000000
 99 002374  177777                          .WORD   177777
100 002375  000000                          .WORD   000000
101 002376  177777                          .WORD   177777
102 002377  140753                PAT7:     .WORD   140753          ; EDC FOR PATTERN 7
103 002400  000001                          .WORD   1               ; B'1011011011011001'
104 002401  133331                          .WORD   133331
105 002402  021147                PAT8:     .WORD   021147          ; EDC FOR PATTERN 8
106 002403  000020                          .WORD   16.             ; B'0101010101010101' AND
107 002404  052525                          .WORD   052525          ; B'1010101010101010'
108 002405  052525                          .WORD   052525          ; IN 3-2-1-1-1
109 002406  052525                          .WORD   052525
110 002407  125252                          .WORD   125252
111 002410  125252                          .WORD   125252
112 002411  125252                          .WORD   125252
113 002412  052525                          .WORD   052525
114 002413  052525                          .WORD   052525
```

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE 30-2
DATA PATTERNS THAT WILL BE WRITTEN TO THE DRIVE

```
115 002414  125252                        .WORD   125252
116 002415  125252                        .WORD   125252
117 002416  052525                        .WORD   052525
118 002417  125252                        .WORD   125252
119 002420  052525                        .WORD   052525
120 002421  125252                        .WORD   125252
121 002422  052525                        .WORD   052525
122 002423  125252                        .WORD   125252
123 002424  041260              PAT9:     .WORD   041260          ; EDC FOR PATTERN 9
124 002425  000001                        .WORD   1               ; B'1101101101101100'
125 002426  155554                        .WORD   155554
126 002427  074075              PAT10:    .WORD   074075          ; EDC FOR PATTERN 10
127 002430  000020                        .WORD   16.             ; B'0010110100101101' AND
128 002431  026455                        .WORD   026455          ; B'1101001011010010'
129 002432  026455                        .WORD   026455          ; IN 3-2-1-1-1
130 002433  026455                        .WORD   026455
131 002434  151322                        .WORD   151322
132 002435  151322                        .WORD   151322
133 002436  151322                        .WORD   151322
134 002437  026455                        .WORD   026455
135 002440  026455                        .WORD   026455
136 002441  151322                        .WORD   151322
137 002442  151322                        .WORD   151322
138 002443  026455                        .WORD   026455
139 002444  151322                        .WORD   151322
140 002445  026455                        .WORD   026455
141 002446  151322                        .WORD   151322
142 002447  026455                        .WORD   026455
143 002450  151322                        .WORD   151322
144 002451  153110              PAT11:    .WORD   153110          ; EDC FOR PATTERN 11
145 002452  000001                        .WORD   1               ; B'0110110110110110'
146 002453  066666                        .WORD   066666
147 002454  046211              PAT12:    .WORD   046211          ; EDC FOR PATTERN 12
148 002455  000020                        .WORD   16.             ; RIPLE ONE
149 002456  000001                        .WORD   000001
150 002457  000002                        .WORD   000002
151 002460  000004                        .WORD   000004
152 002461  000010                        .WORD   000010
153 002462  000020                        .WORD   000020
154 002463  000040                        .WORD   000040
155 002464  000100                        .WORD   000100
156 002465  000200                        .WORD   000200
157 002466  000400                        .WORD   000400
158 002467  001000                        .WORD   001000
159 002470  002000                        .WORD   002000
160 002471  004000                        .WORD   004000
161 002472  010000                        .WORD   010000
162 002473  020000                        .WORD   020000
163 002474  040000                        .WORD   040000
164 002475  100000                        .WORD   100000
165 002476  121147              PAT13:    .WORD   121147          ; EDC FOR PATTERN 13
166 002477  000020                        .WORD   16.             ; RIPLE ZERO
167 002500  177776                        .WORD   177776
168 002501  177775                        .WORD   177775
169 002502  177773                        .WORD   177773
170 002503  177767                        .WORD   177767
171 002504  177757                        .WORD   177757
```

DATA PATTERNS THAT WILL BE WRITTEN TO THE DRIVE

```
    172 002505  177737                        .WORD   177737
    173 002506  177677                        .WORD   177677
    174 002507  177577                        .WORD   177577
    175 002510  177377                        .WORD   177377
    176 002511  176777                        .WORD   176777
    177 002512  175777                        .WORD   175777
    178 002513  173777                        .WORD   173777
    179 002514  167777                        .WORD   167777
    180 002515  157777                        .WORD   157777
    181 002516  137777                        .WORD   137777
    182 002517  077777                        .WORD   077777
    183 002520  125677              PAT14:    .WORD   125677          ; EDC FOR PATTERN 14
    184 002521  000003                        .WORD   3               ; B'1101101101101101'
    185 002522  155555                        .WORD   155555          ; B'1011011011011011' PATTERNS 14 AND 15
    186 002523  133333                        .WORD   133333          ; B'1101101101101101' MAKE TEST 4 FORMATTER
    187 002524  155555                        .WORD   155555          ; REPEATED            COMPATIBLE
    188 002525  030206              PAT15:    .WORD   030206          ; EDC FOR PATTERN 15
    189 002526  000003                        .WORD   3               ; B'1101101101101101'
    190 002527  155555                        .WORD   155555          ; B'1011011011011011'
    191 002530  133333                        .WORD   133333          ; B'0110110110110110'
    192 002531  066666                        .WORD   066666          ; REPEATED
```

```
                                        .SBTTL  ROOT    - MAIN DRIVING MODULE OF TEST 4

002532  104205  007702        ROOT:   MOV     #FIRSTU,R5              ; R5 POINTS TO FIRST UNIT TO BE TESTED
002534  114007                        CLR     R0                      ; COUNT OF DROPPED UNITS IS SET TO ZERO
002535  104155                ROOTLP: MOV     (R5),R5                 ; R5 POINTS TO NEXT UNIT IN CHAIN
002536                                ASSUME  U.NEXT,0
002536  104651  000046                MOV     U.PARM(R5),R1           ; GET UNIT PARAMETERS
002540  072560                        BMI     NOUNIT                  ; IF UNIT DROPPED, BRANCH
002541                                ASSUME  DROP,100000             ; ASSUME DROP IS SIGN BIT
002541  115000  002223                TST     M.PARM                  ; SEE IF IN PROCESS OF INITIAL WRITE
002543  032547                        BPL     1$                      ; IF NOT, BRANCH
002544                                ASSUME  IWIPRG,100000           ; ASSUME IWIPRG IS SIGN BIT
002544  102201  040000                BIT     #INITW,R1               ; SEE IF THIS UNIT IS BEING INITALLY WRITTEN
002546  012560                        BEQ     NOUNIT                  ; IF NOT, BRANCH
002547  022577                1$:     CALL                    ; CALL          R
002550  104657  000046                MOV     U.PARM(R5),R0           ; SEE IF JUST DROPPED
002552  032556                        BPL     2$                      ; IF NOT, BRANCH
002553                                ASSUME  DROP,100000
002553  104203  001650                MOV     #CR.DIS,R3              ; POINT TO DISCONNECT
002555  021173                        CALL    TALK                    ; SEND
002556  114007                2$:     CLR     R0                      ; COUNT OF DROPPED UNITS IS SET TO ZERO
002557  002535                        BR      ROOTLP                  ; BRANCH TO BOTTOM OF ROOT
002560  115407                NOUNIT: INC     R0                      ; INCREMENT COUNT OF DROPPED UNITS
002561  106207  000004                CMP     #4,R0                   ; SEE IF FOUR (ALL) UNITS HAVE BEEN DROPPED
002563  052535                        BNE     ROOTLP                  ; IF NOT, BRANCH
002564  115000  002223                TST     M.PARM                  ; SEE IF INITIAL WRITE WAS IN PROGRESS
002566  032573                        BPL     STOP                    ; IF NOT, END TEST4
002567                                ASSUME  IWIPRG,100000           ; ASSUME IWIPRG IS SIGN BIT
002567  103200  100000  002223        BIC     #IWIPRG,M.PARM          ; CLEAR INITIAL WRITE BIT
002572  002532                        BR      ROOT                    ; NOW START TESTING
002573  104207  060016        STOP:   MOV     #DONE,R0                ; MOVE DONE TO HOST COMMUNICATION AREA
002575  021053                        CALL    HOSTRQ                  ; SENT DONE MESSAGE TO HOST
002576  060021                        XFC     EXIT                    ; EXIT DIAGNOSTIC MACHINE MODE
```

```
                                .SBTTL          OVERLAY DRIVER FOR TEST 4
002577
                                ;
                                ;               R CALLS THE VARIOUS MODULES, USING THE ADDRESS RETURNED
                                ;       IN R0 BY THE PREVIOUS MODULE.  R0 HOLDS THE NEXT ADDRESS TO VECTOR
                                ;       TO. IF R1 IS ZERO, AN IMMIDATE CALL IS MADE TO THE NEXT ROUTINE.
                                ;       IF R2 IS NONZERO, AN ERROR WAS ENCOUNTERED, AND THE ERROR COUNT IS
                                ;       INCREMENTED.  THE ERROR IS THEN REPORTED TO THE HOST.
                                ;
002577  104657  000013          MOV     U.NFUN(R5),R0   ; LOAD R0 WITH NEXT FUNCTION ADDRESS
002601
```

```
                                        .SBTTL  DRPTST - SEE THAT AT LEAST ONE SUBUNIT IS ACTIVE ON A UNIT
                                        ;DRPTST
                                        ;
                                        ;       DROP TEST MAKES SURE AT LEAST ONE ACTIVE SUBUNIT IS ACTIVE ON THE UNIT
                                        ;       IF SO, R4 POINTS TO SUBUNIT PARAMETER BLOCK, U.SUBU IS UPDATED TO
                                        ;       THE SUBUNIT OFFSET, AND R3 IS RETURNED POSITIVE.  IF NOT, R3 IS
                                        ;       RETURNED NEGATIVE AND THE ENTIRE UNIT IS DROPPED
                                        ;
 002601                                         PUSH    R0              ; SAVE NEXT MODULE ADDRESS
 002601  100467                                                                 MOV R0,-(SP)
 002602  104657  000046                         MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
 002604  102207  001000                         BIT     #FTIME,R0       ; SEE IF FIRST TIME THROUGH
 002606  012673                                BEQ     1$              ; IF NOT, BRANCH
 002607  024227                                CALL    GOSEEK          ; IF FIRST TIME, FORCE A SEEK
 002610  103207  015610                         BIC     #RBNBN!REVEC!FTIME!DIREC!WCHECK!NEWSUB,R0
 002612  100657  000046                         MOV     R0,U.PARM(R5)   ; SAVE
 002614  104057                                MOV     R5,R0           ; R0 WILL POINT AT SUBUNIT POINTERS
 002615  115407                                INC     R0              ; R0 POINTS AT SUBUNIT POINTERS
 002616                                         ASSUME  U.SUBP,1
 002616  114002                                CLR     R2              ; UP TO 4 SUBUNITS ON THIS UNIT
 002617  100652  000024                         MOV     R2,U.CSEC(R5)   ; ZERO ALL TRACK COUNTS SO THE TEST AT THE
 002621  100652  000021                         MOV     R2,U.NSEC(R5)   ; BEGINNING OF SETUP IS SATISIFIED SO CONTROL
 002623  100652  000023                         MOV     R2,U.TSEC(R5)   ; DROPS THROUGH TO SETUP THE NEXT OPERATION
 002625  104273                        11$:    MOV     (R0)+,R3        ; GET SUBUNIT POINTER
 002626  072637                                BMI     12$             ; IF NO SUBUNIT, BRANCH
 002627  104133                                MOV     (R3),R3         ; GET SUBUNIT PARAMETERS
 002630                                         ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
 002630                                         ASSUME  DROP,100000     ; ASSUME DROP IS SIGN BIT
 002630  072637                                BMI     12$             ; IF DROPPED, BRANCH
 002631  115000  002223                         TST     M.PARM          ; SEE IF INTIIAL WRITE IN PROGRESS
 002633  032644                                BPL     13$             ; IF NOT, BRANCH
 002634                                         ASSUME  IWIPRG,100000   ; ASSUME IWIPRG IS SIGN BIT
 002634  102203  040000                         BIT     #INITW,R3       ; SEE IF .HIS SUBUNIT IS TO BE INITIALLY WRITTEN
 002636  052644                                BNE     13$             ; IF SO, BRANCH
 002637  115402                        12$:    INC     R2              ; INCREMENT COUNT
 002640  106202  000003                         CMP     #3,R2           ; SEE IF COUNT EXHAUSTED
 002642  032625                                BPL     11$             ; IF NOT, BRANCH
 002643  003037                                BR      4$              ; IF SO, DROP ENTIRE UNIT
 002644  100652  000050                13$:    MOV     R2,U.SUBU(R5)   ; SAVE SUBUNIT OFFSET
 002646  104024                                MOV     R2,R4           ; MOVE TO SUBUNIT POINTER REGISTER
 002647  105054                                ADD     R5,R4           ; ADD UNIT POINTER
 002650  105204  000001                         ADD     #U.SUBP,R4      ; ADD SUBUNIT POINTER OFFSET
 002652  104144                                MOV     (R4),R4         ; R4 NOW POINTS TO SUBUNIT
 002653  023720                                CALL    RECOVR          ; SET UP UNIT FOR RUNNING
 002654  104207  104000                         MOV     #ERMASK!SEKREQ,R0 ; DISABLE RECOVERY, FORCE A SEEK
 002656  100657  000047                         MOV     R0,U.RCOV(R5)   ; SAVE IN RECOVERY WORD
 002660  104207  000010                         MOV     #NUML2S,R0      ; GET NUMBER OF LEVEL 2 COMMANDS
 002662  104051                                MOV     R5,R1           ; R1 POINTS TO UNIT PARAMETERS
 002663  105201  000035                         ADD     #U.SDI2,R1      ; R1 POINTS TO SDI LEVEL 2 ERROR COUNT AREA
 002665  114002                                CLR     R2              ; SET UP TO CLEAR AREA
 002666  100212                        14$:    MOV     R2,(R1)+        ; CLEAR
 002667  117407                                DEC     R0              ; DECREMENT COUNT
 002670  052666                                BNE     14$             ; IF INCOMPLETE, BRANCH
 002671  114003                                CLR     R3              ; PROCESS THIS UNIT
 002672  003045                                BR      5$              ; EXIT
 002673  104054                        1$:     MOV     R5,R4           ; GET POINTER TO UNIT DATASTRUCTURE
 002674  115404                                INC     R4              ; POINT AT SUBUNIT POINTERS
```

```
                                        ASSUME  U.SUBP,1
                                        ADD     U.SUBU(R5),R4   ; R4 POINTS AT SPECIFIC SUBUNIT POINTER
                                        MOV     (R4),R4         ; R4 POINTS AT SUBUNIT PARAMETERS
                                        BMI     25$             ; NO SUBUNIT (SOMETIMES HAPPENS IN INITIALIZATION)
                                        MOV     (R4),R1         ; GET SUBUNIT PARAMETERS
                                        ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
                                        BIT     #NEWSUB,R0      ; SEE IF SUPPOSED TO GO TO NEXT SUBUNIT
                                        BEQ     6$              ; IF NOT, BRANCH
                                        BIC     #NEWSUB,R0      ; CLEAR NEWSUB BIT
                                        MOV     R0,U.PARM(R5)   ; SAVE
                                        CLR     R2              ; FOR CEARING FOLLOWING WORDS
                                        MOV     R2,U.CSEC(R5)   ; ZERO ALL TRACK COUNTS SO THE TEST AT THE
                                        MOV     R2,U.NSEC(R5)   ; BEGINNING OF SETUP IS SATISIFIED SO CONTROL
                                        MOV     R2,U.TSEC(R5)   ; DROPS THROUGH TO SETUP THE NEXT OPERATION
                                        TST     M.PARM          ; SEE IF INITIAL WRITE IS IN PROGRESS
                                        BPL     2$              ; IF NOT, BRANCH
                                        ASSUME  IWIPRG,100000   ; ASSUME IWIPRG IS SIGN BIT
                                        BIC     #INITW,R1       ; CLEAR INITIAL WRITE BIT
                                        MOV     R1,(R4)         ; SAVE
                                        ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
                                        MSSG    1               ; REPORT INITIAL WRITE COMPLETE
                                                                        MOV     #MS1,HRQ.02
                                                                                MOV R0,-(SP)
                                                                        MOV     U.UNUM(R5),HRQ.01
                                                                        ADD     U.SUBU(R5),HRQ.01
                                                                        MOV     #MESSAG,R0
                                                                        CALL    HOSTRQ
                                                                                MOV (SP)+,R0
                                        MOV     U.SUBU(R5),R2   ; GET ACTIVE SUBUNIT POINTER
                                        CALL    CHKUP           ; SEE IF ANY MORE UNITS TO INITIAL WRITE
                                        BR      5$              ; BRANCH
                                6$:     CLR     R3              ; IF ACTIVE, FLAG TO PROCESS
                                        TST     R1              ; SEE IF UNIT ACTIVE
                                        ASSUME  DROP,100000     ; ASSUME DROP IS SIGN BIT
                                        BPL     7$              ; IF ACTIVE, BRANCH
                                2$:     BIT     #       NITW,R1         ; SEE IF         AL SEEKS
                                        BNE     10$             ; IF SO, BRANCH
                                ;RNDSUB
                                ;
                                ;       GET NEXT SUBUNIT TO CHECK RANDOMLY
                                ;
                                25$:    MOV     #4,R0           ; SUBUNIT COUNT
                                        MOV     #U.SUBP,R3      ; R3 WILL POINT TO SUBUNIT POINTERS
                                        ADD     R5,R3           ; R3 POINTS TO SUBUNIT POINTERS
                                        MOV     U.SUBU(R5),R2   ; R2 IS SUBUNIT NUMBER
                                21$:    INC     R2              ; GO TO NEXT UNIT
                                        CMP     #3,R2           ; SEE IF GREATER THAN 3
                                        BPL     22$             ; IF NOT, BRANCH
                                        CLR     R2              ; NOW ON SUBUNIT 0
                                22$:    MOV     R3,R4           ; R4 POINTS TO SUBUNIT POINTERS
                                        ADD     R2,R4           ; R4 NOW POINTS TO A SPECIFIC SUBUNIT POINTER
                                        MOV     (R4),R4         ; R4 NOW POINTS TO SUBUNIT PARAMETERS
                                        BMI     23$             ; IF NO SUBUNIT, BRANCH
                                        MOV     (R4),R1         ; GET SUBUNIT PARAMETERS
                                        ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
                                        BPL     24$             ; IF ACTIVE, BRANCH
                                23$:    DEC     R0              ; DECREMENT COUNT
```

```
 107 003001  052765                        BNE     21$             ; IF INCOMPLETE, BRANCH
 108 003002  104653  000046                MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
 109 003004  101203  100000                BIS     #DROP,R3        ; SET DROP BIT
 110 003006  100653  000046                MOV     R3,U.PARM(R5)   ; SAVE
 111 003010  003045                        BR      5$              ; EXIT
 112 003011  100652  000050          24$:  MOV     R2,U.SUBU(R5)   ; SAVE SUBUNIT NUMBER
 113 003013  114003                        CLR     R3              ; TEST THIS SUBUNIT IMMIDATELY
 114 003014  003045                        BR      5$              ; EXIT
 115 003015  104652  000050          10$:  MOV     U.SUBU(R5),R2   ; R2 IS SUBUNIT NUMBER (0-3)
 116 003017  102207  010000                BIT     #DIREC,R0       ; SEE IF GOING UP OR DOWN
 117 003021  053031                        BNE     3$              ; IF DOWN, BRANCH
 118 003022  024134                        CALL    CHKUP           ; FIND NEXT UPPER SUBUNIT
 119 003023  115007                        TST     R0              ; FIND ONE?
 120 003024  013045                        BEQ     5$              ; IF SO, BRANCH
 121 003025  024104                        CALL    CHKDN           ; SEARCH DOWN FOR ONE
 122 003026  115007                        TST     R0              ; FIND ONE?
 123 003027  013045                        BEQ     5$              ; IF SO, BRANCH
 124 003030  003037                        BR      4$              ; DROP UNIT
 125 003031  024104                  3$:   CALL    CHKDN           ; LOOK DOWN FOR SUBUNIT
 126 003032  115007                        TST     R0              ; FIND ONE?
 127 003033  013045                        BEQ     5$              ; IF SO, BRANCH
 128 003034  024134                        CALL    CHKUP           ; LOOK UP FOR ONE
 129 003035  115007                        TST     R0              ; FIND ONE?
 130 003036  013045                        BEQ     5$              ; IF SO, BRANCH
 131 003037  104653  000046          4$:   MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
 132 003041  101203  100000                BIS     #DROP,R3        ; DROP UNIT
 133 003043  100653  000046                MOV     R3,U.PARM(R5)   ; SAVE
 134 003045  115003                  5$:   TST     R3              ; SEE IF THIS SUBUNIT TO BE TESTED
 135 003046  053052                        BNE     7$              ; IF NOT, BRANCH
 136 003047  104200  177777  002217        MOV     #-1,SCR1        ; FLAG AS NEW SUBUNIT
 137 003052                          7$:   POP     R0              ; RESTORE R0
     003052  104267                                                MOV (SP)+,R0
 138 003053  115003                        TST     R3              ; SEE IF ACTIVE SUBUNIT EXISTS
 139 003054  053715                        BNE     NOSUB           ; IF SO, BRANCH
 140 003055  104070  002222                MOV     R0,LSTOVL       ; SAVE THIS OVERLAY NUMBER
 141 003057  106207  000100                CMP     #100,R0         ; SEE IF THE ROUTINE IS ON AN OVERLAY
 142 003061  073214                        BMI     GO4IT           ; IF NO1, BRANCH
 143 003062  104071                        MOV     R0,R1           ; SAVE THE OVERLAY NUMBER
 144 003063  105077                        ADD     R0,R0           ; MULTIPLY THE OVERLAY NUMBER BY 4
 145 003064  105077                        ADD     R0,R0
 146 003065  104672  004266                MOV     OTABLE(R0),R2   ; GET THE OVERLAY AREA NUMBER
 147 003067  105022                        ADD     R2,R2           ; MULTIPLY THE AREA NUMBER BY 2
 148 003070  106621  004254                CMP     PTABLE(R2),R1   ; SEE IF THE OVERLAY REQUESTED IS IN MEMORY
 149 003072  013175                        BEQ     LOADED          ; IF SO, BRANCH
```

```
                                .SBTTL  OVRLAY - OVERLAY PROCESS FOR BRINGING IN OVERLAYS IF NEEDED
                                ;OVRLAY
                                ;
                                ;       BRING IN THE OVERLAY
                                ;
 6 003073                       PUSH    <R5,R0,R1,R2,R3>        ; SAVE REGISTERS
   003073  100465                                                               MOV R5,-(SP)
   003074  100467                                                               MOV R0,-(SP)
   003075  100461                                                               MOV R1,-(SP)
   003076  100462                                                               MOV R2,-(SP)
   003077  100463                                                               MOV R3,-(SP)
 7 003100  104205  000003       MOV     #3,R5                   ; TRY TO READ 3 TIMES
 8 003102  104671  004271 41$:  MOV     OTABLE+3(R0),R1 ; GET THE HI ORDER BITS OF UNIBUS ADDRESS
 9 003104  103201  177774       BIC     #177774,R1              ; CLEAR UNUSED BITS
10 003106  104623  004255       MOV     PTABLE+1(R2),R3 ; POINT TO WHERE TO LOAD OVERLAY
11 003110  104672  004271       MOV     OTABLE+3(R0),R2 ; GET NUMBER OF WORDS TO TRANSFER
12 003112  110602               ROR     R2                      ; ROTATE BITS TO CORRECT POSITION
13 003113  110602               ROR     R2
14 003114  103202  140000       BIC     #140000,R2              ; CLEAR UNUSED BITS
15 003116  104677  004270       MOV     OTABLE+2(R0),R0 ; GET LO ORDER UNIBUS ADDRESS
16 003120  060013               XFC     UREAD                   ; READ THE OVERLAY INTO UDA MEMORY
17 003121  101071               BIS     R0,R1                   ; ADD THE ERROR REGISTERS TOGETHER
18 003122  013166               BEQ     42$                     ; IF NO ERRORS, BRANCH
19 003123                       POP     <R3,R2,R1,R0>   ; RESTORE THE REGISTERS
   003123  104263                                                               MOV (SP)+,R3
   003124  104262                                                               MOV (SP)+,R2
   003125  104261                                                               MOV (SP)+,R1
   003126  104267                                                               MOV (SP)+,R0
20 003127                       PUSH    <R0,R1,R2,R3>   ; SAVE THE REGISTERS
   003127  100467                                                               MOV R0,-(SP)
   003130  100461                                                               MOV R1,-(SP)
   003131  100462                                                               MOV R2,-(SP)
   003132  100463                                                               MOV R3,-(SP)
21 003133  117405               DEC     R5                      ; DECREMENT COUNT
22 003134  053102               BNE     41$                     ; IF INCOMPLETE, BRANCH
23 003135                       DEVFTL  33,<R1,#SER39>  ; UNABLE TO READ FROM HOST MEMORY
   003135  104200  002455  001107                                       MOV     #ER33,HRQ.04
   003140  104010  001110                                               MOV     R1,HRQ.05
   003142  104200  002502  001111                                       MOV     #SER39,HRQ.06
   003145  104202  047701                                               MOV     #33!FTLDEV+4000.,R2
   003147  104020  001105                                               MOV     R2,HRQ.02
   003151  104200  003151  001104                                       MOV     #.,HRQ.01
   003154  104200  060014  001103                                       MOV     #ERRMC,HRQ.RQ
24 003157  104200  177777  001106       MOV     #177777,HRQ.03  ; FLAG AS NOT ASSOCIATED WITH ANY UNIT
25 003162  104307  001103               MOV     HRQ.RQ,R0               ; SET UP TO REPORT ERROR
26 003164  021053               CALL    HOSTRQ                  ; REPORT
27 003165  002573               BR      STOP                    ; REPORT THAT PROGRAM IS FINISHED
28 003166                  42$: POP     <R3,R2,R1,R0,R5>        ; RESTORE THE REGISTERS
   003166  104263                                                               MOV (SP)+,R3
   003167  104262                                                               MOV (SP)+,R2
   003170  104261                                                               MOV (SP)+,R1
   003171  104267                                                               MOV (SP)+,R0
   003172  104265                                                               MOV (SP)+,R5
29 003173  100621  004254       MOV     R1,PTABLE(R2)   ; MARK THE OVERLAY AS IN MEMORY
30 003175  104677  004267 LOADED: MOV     OTABLE+1(R0),R0 ; GET THE NUMBER OF OVERLAY AREAS DESTROYED
31 003177  104201  177777       MOV     #-1,R1                  ; R1 CONTAINS THE 'EMPTY AREA' FLAG
32 003201                       PUSH    R2                      ; SAVE R2
```

```
                003201  100462                                                                          MOV R2,-(SP)
     33         003202  117407          GONE:   DEC     R0              ; DECREMENT THE DESTROYED COUNT
     34         003203  073211                  BMI     PLOAD           ; IF ALL NECESSARY AREAS MARKED DESTROYED, BRANCH
     35         003204  105202  000002          ADD     #2,R2           ; POINT TO NEXT OVERLAY AREA
     36         003206  100621  004254          MOV     R1,PTABLE(R2)   ; MARK AREA AS DESTROYED
     37         003210  003202                  BR      GONE            ; GO TO TOP OF LOOP
     38         003211          PLOAD:  POP     R2                      ; RESTORE R2
                003211  104262                                                                          MOV (SP)+,R2
     39         003212  104627  004255          MOV     PTABLE+1(R2),R0 ; GET OVERLAY ADDRESS
     40         003214  104652  000047  GO4IT:  MOV     U.RCOV(R5),R2   ; GET ERROR RECOVERY WORD
     41         003216  073227                  BMI     2$              ; IF NO RECOVERY, BRANCH
     42         003217                          ASSUME  ERMASK,100000   ; ASSUME ERMASK SIGN BIT
     43         003217  020746                  CALL    RTDS            ; BEFORE OPERATION, SEE IF ONLINE
     44         003220  115002                  TST     R2              ; SEE IF ERROR OCCURRED
     45         003221  053225                  BNE     1$              ; IF SO, BRANCH
     46         003222  102201  000102          BIT     #AVAIL!ATTN,R1  ; SEE IF AVAILABLE OR ATTENTION ASSERTED
     47         003224  013227                  BEQ     2$              ; IF NOT, BRANCH
     48         003225  024221          1$:     CALL    GORTRY          ; RETRY THIS MODULE
     49         003226  003231                  BR      JMPRET          ; RECOVER
     50         003227          2$:
```

```
      1                                 .SBTTL  JMPRET - THIS IS THE LOCATION THAT VECTORS TO/FROM ALL OVERLAYS
      2 003227                                  PUSH    R0              ; PUSH THE BRANCH ADDRESS ONTO THE STACK
        003227  100467                                                                      MOV R0,-(SP)
      3 003230  000000                          RETURN                  ; VECTOR TO MODULE
      4 003231                          JMPRET:
```

```
      1                                 .SBTTL  CORECT - CORRECT THE ERRORS
      2                                 ;CORECT
      3                                 ;
      4                                 ;       CORRECT THE ERRORS AND RECOVER
      5                                 ;
      6 003231                                  PUSH    <R1,R0>         ; SAVE NEXT OVERLAY AND CALL TYPE
        003231  100461                                                                  MOV R1,-(SP)
        003232  100467                                                                  MOV R0,-(SP)
      7 003233  114000  002226                  CLR     ERMODE          ; CLEAR ERROR MODES
      8 003235  115002                          TST     R2              ; SEE IF ERROR OR MBYTE REPORT
      9 003236  053331                          BNE     5$              ; IF SO, BRANCH
     10 003237  104652  000047                  MOV     U.RCOV(R5),R2   ; GET ERROR RECOVERY WORD
     11 003241  073711                          BMI     90$             ; IF RECOVERY DISABLED, BRANCH
     12 003242                                  ASSUME  ERMASK,100000   ; ASSUME RECOVERY MASK IS SIGN BIT
     13 003242  020746                          CALL    RTDS            ; GET REAL TIME DRIVE STATE
     14 003243  115002                          TST     R2              ; SEE IF ERROR OCCURRED
     15 003244  053361                          BNE     70$             ; IF SO, REPORT
     16 003245  102201  000100                  BIT     #AVAIL,R1       ; SEE IF AVAILABLE ASSERTED
     17 003247  013277                          BEQ     2$              ; IF NOT, BRANCH
     18 003250  102201  000002                  BIT     #ATTN,R1        ; SEE IF SPINABLE
     19 003252  053273                          BNE     1$              ; IF SO, BRANCH
     20 003253                                  DEVFTL  13              ; REPORT UNSPINABLE DRIVE
        003253  104200  001411  001107                                                  MOV     #ER13,HRQ.04
        003256  104202  047655                                                          MOV     #13!FTLDEV+4000.,R2
        003260  104020  001105                                                          MOV     R2,HRQ.02
        003262  104200  003262  001104                                                  MOV     #.,HRQ.01
        003265  104200  060014  001103                                                  MOV     #ERRMC,HRQ.RQ
     21 003270                                  ENDERR  0
        003270  114000  002230                                                          CLR     ERRPOS          ; CLEAR THE POSITION
     22 003272  003361                          BR      70$             ; REPORT
     23 003273  101200  000100  002226  1$:     BIS     #AVAIL,ERMODE   ; AVAIL ERROR
     24 003276  003302                          BR      3$              ; BRANCH
     25 003277  102201  000002          2$:     BIT     #ATTN,R1        ; SEE IF ATTENTION IS ASSERTED
     26 003301  013711                          BEQ     90$             ; IF NOT, BRANCH
     27 003302  101200  000002  002226  3$:     BIS     #ATTN,ERMODE    ; FLAG AS ATTENTION ERROR
     28 003305                                  SOFTER  2               ; ATTENTION ASSERTED UNEXPECTEDLY
        003305  104200  000117  001107                                                  MOV     #ER2,HRQ.04
        003310  104202  147642                                                          MOV     #2!ERSOFT+4000.,R2
        003312  104020  001105                                                          MOV     R2,HRQ.02
        003314  104200  003314  001104                                                  MOV     #.,HRQ.01
        003317  104200  060013  001103                                                  MOV     #ERRMES,HRQ.RQ
     29 003322                                  ENDERR
        003322  104200  000051  001110                                                  MOV     #SER22,HRQ.05
        003325  104200  000006  002230                                                  MOV     #5+1,ERRPOS     ; SET THE POSITION
     30 003330  003364                          BR      6$              ; BRANCH
```

```
      1 003331  106200  060011  001103  5$:     CMP     #T4MXFR,HRQ.RQ  ; SEE IF MEGABIT TRANSFER REPORT
      2 003334  053352                          BNE     40$             ; IF NOT, BRANCH
      3 003335  104650  000063  001104          MOV     U.UNUM(R5),HRQ.01       ; MOVE STARTING UNIT NUMBER TO COM AREA
      4 003340  105650  000050  001104          ADD     U.SUBU(R5),HRQ.01       ; ADD OFFSET
      5 003343  104307  001103                  MOV     HRQ.RQ,R0       ; SETUP FOR COM
      6 003345  021053                          CALL    HOSTRQ          ; REPORT
      7 003346  115000  001104                  TST     HRQ.01          ; SEE IF DROPPED
      8 003350  073656                          BMI     80$             ; IF SO, BRANCH
      9 003351  003711                          BR      90$             ; IF NOT, BRANCH
```

```
 1 003352  104302  001105          40$:    MOV     HRQ.02,R2       ; GET ERROR TYPE
 2 003354  103202  037777                  BIC     #37777,R2       ; CLEAR ERROR NUMBER
 3 003356  106202  040000                  CMP     #FTLDEV,R2      ; SEE IF DEVICE FATAL
 4 003360                                  ASSUME  FTLDEV,040000   ; ASSUME IN HI 2 BITS
 5 003360  053364                          BNE     6$              ; IF NOT, BRANCH
 6 003361  101200  100000  002226  70$:    BIS     #DROP,ERMODE    ; ERROR CAUSED THE ENTIRE DEVICE TO BE DROPPED
 7 003364  104307  002230          6$:     MOV     ERRPOS,R0       ; GET ERROR POSITION
 8 003366                                  ASSUME  DROP,100000     ;
 9 003366  013543                          BEQ     20$             ; IF NO REPORTING, BRANCH
10 003367  103207  100000                  BIC     #VALID,R0       ; CLEAR 'ALLREADY VALID' BIT
11 003371  105207  001103                  ADD     #HRQ.RQ,R0      ; POINT TO OUTPUT BUFFER
12 003373  020775                          CALL    RTDSL           ; GET REAL TIME DRIVE STATE
13 003374  100271                          MOV     R1,(R0)+        ; MOVE TO OUTPUT BUFFER
14 003375                                  PUSH    <R0,R4>         ; SAVE R0 AND SUBUNIT POINTER
   003375  100467                                                                  MOV R0,-(SP)
   003376  100464                                                                  MOV R4,-(SP)
15 003377  104651  000047                  MOV     U.RCOV(R5),R1   ; GET RECOVERY STATUS
16 003401  102201  040000                  BIT     #DRINIT,R1      ; SEE IF DRIVE INITIALIZED
17 003403  013454                          BEQ     4$              ; IF NOT, BRANCH
18 003404  103201  040000                  BIC     #DRINIT,R1      ; DRIVE NOT INITIALIZED
19 003406  100651  000047                  MOV     R1,U.RCOV(R5)   ; SAVE
20                                  ;       WAIT UNTIL STATE WILL BE VALID
21 003410  104201  001400                  MOV     #1400,R1        ; SET UP TIMEOUT
22 003412  117401                  33$:    DEC     R1              ; DECREMENT DELAY
23 003413  053412                          BNE     33$             ; IF INCOMPLETE, LOOP
24 003414  104651  000011                  MOV     U.MSTO(R5),R1   ; GET MASTER SEEK TIMEOUT
25 003416  100651  000005                  MOV     R1,U.TIMH(R5)   ; SAVE
26 003420  114001                          CLR     R1              ; FOR LOW ORDER TIMEOUT
27 003421  100651  000006                  MOV     R1,U.TIML(R5)   ; SAVE
28 003423  104201  000310          31$:    MOV     #200.,R1        ; INNER LOOP TIMEOUT
29 003425  117401                  32$:    DEC     R1              ; DECREMENT COUNT
30 003426  053425                          BNE     32$             ; IF INCOMPLETE, BRANCH
31 003427  020775                          CALL    RTDSL           ; GET REAL TIME DRIVE STATE
32 003430  115002                          TST     R2              ; SEE IF ERROR OCCURRED
33 003431  053517                          BNE     24$             ; IF SO, BRANCH
34 003432  102201  000001                  BIT     #RCVRDY,R1      ; SEE IF RECEIVER READY IS ASSERTED
35 003434  053454                          BNE     4$              ; IF SO, BRANCH
36 003435  104651  000006                  MOV     U.TIML(R5),R1   ; GET TIMEOUT
37 003437  107201  000001                  SUB     #1,R1           ; DECREMENT TIMEOUT
38 003441  100651  000006                  MOV     R1,U.TIML(R5)   ; SAVE
39 003443  043423                          BCC     31$             ; LOOP IF INCOMPLETE
40 003444  104651  000005                  MOV     U.TIMH(R5),R1   ; GET HI ORDER TIMEOUT
41 003446  107201  000001                  SUB     #1,R1           ; DECREMENT TIMEOUT
42 003450  100651  000005                  MOV     R1,U.TIMH(R5)   ; SAVE
43 003452  043423                          BCC     31$             ; IF TIMOUT UNEXPIRED, BRANCH
44 003453  003517                          BR      24$             ; ERROR
45 003454  115000  002230          4$:     TST     ERRPOS          ; SEE IF STATUS VALID
46 003456  073531                          BMI     26$             ; IF SO, BRANCH
47 003457                                  ASSUME  VALID,100000    ; ASSUME VALID BIT IS MSB
48 003457  020775                          CALL    RTDSL           ; GET REAL TIME STATE
49 003460  115002                          TST     R2              ; SEE IF STATE VALID
50 003461  053517                          BNE     24$             ; IF INVALID, BRANCH
51 003462  102201  000101                  BIT     #RCVRDY!AVAIL,R1        ; SEE IF RECEIVER READY/AVAILABLE
52 003464  013517                          BEQ     24$             ; IF NOT, BRANCH
53 003465  104204  001750                  MOV     #MAXSND,R4      ; SET UP TIMEOUT
54 003467  104203  001617                  MOV     #CR.GST,R3      ; POINT TO GET STATUS COMMAND
55 003471  104652  000025                  MOV     U.MASK(R5),R2   ; GET UNIT SDI SELECT MASK
```

```
     56 003473  104137                  21$:    MOV     (R3),R0         ; POINTS TO SDI COMMAND BUFFER
     57 003474                                  ASSUME  L2.OPC,0
     58 003474  104631  000001                  MOV     L2.SLN(R3),R1   ; LOAD BYTE COUNT
     59 003476  060004                          XFC     SEND            ; SEND SDI COMMAND
     60 003477  115001                          TST     R1              ; SEE IF SDI COMMAND SENT SUCESSFULLY
     61 003500  013504                          BEQ     22$             ; IF SO, BRANCH
     62 003501  117404                          DEC     R4              ; DECREMENT TIMEOUT
     63 003502  053473                          BNE     21$             ; IF TIMEOUT INCOMPLETE, BRANCH
     64 003503  003517                          BR      24$             ; BRANCH
     65 003504  104654  000033          22$:    MOV     U.SDIS(R5),R4   ; R4 HAS SHORT TIMEOUT
     66 003506  104207  001702          23$:    MOV     #ST,R0          ; POINT TO RECEIVE BUFFER
     67 003510  104631  000002                  MOV     L2.RLN(R3),R1   ; NUMBER OF WORDS IN RESPONSE
     68 003512  060005                          XFC     RCV             ; RECEIVE SDI RESPONSE
     69 003513  115001                          TST     R1              ; SEE IF SDI RESPONSE RECEIVED SUCESSFULLY
     70 003514  013531                          BEQ     26$             ; IF SO, BRANCH
     71 003515  117404                          DEC     R4              ; DECREMENT TIMEOUT VALUE
     72 003516  053506                          BNE     23$             ; IF TIMEOUT UNEXPIRED, BRANCH
     73 003517                          24$:    POP     <R4,R0>         ; RESTORE R4
        003517  104264                                                                  MOV (SP)+,R4
        003520  104267                                                                  MOV (SP)+,R0
     74 003521  104203  177777                  MOV     #-1,R3          ; R3 CONTAINS 'UNABLE TO GET STATUS'
     75 003523  104202  000007                  MOV     #7,R2           ; R2 CONTAINS COUNT
     76 003525  100273                  25$:    MOV     R3,(R0)+        ; MOVE TO OUTPUT BUFFER
     77 003526  117402                          DEC     R2              ; DECREMENT COUNT
     78 003527  053525                          BNE     25$             ; IF COUNT INCOMPLETE, BRANCH
     79 003530  003560                          BR      8$              ; BRANCH
     80 003531                          26$:    POP     <R4,R0>         ; RESTORE R4
        003531  104264                                                                  MOV (SP)+,R4
        003532  104267                                                                  MOV (SP)+,R0
     81 003533  104201  001711                  MOV     #ST+7,R1        ; R1 POINTS TO AFTER STATUS BUFFER
     82 003535  104202  000007                  MOV     #7,R2           ; R2 HAS COUNT
     83 003537  104413                  27$:    MOV     -(R1),R3        ; MOVE STATUS WORD TO R3
     84 003540  100273                          MOV     R3,(R0)+        ; MOVE TO OUTPUT BUFFER
     85 003541  117402                          DEC     R2              ; DECREMENT COUNT
     86 003542  053537                          BNE     27$             ; BRANCH IF COUNT INCOMPLETE
     87 003543  102200  000002  002226  20$:    BIT     #ATTN,ERMODE    ; SEE IF UNEXPECTED ATTENTION
     88 003546  013560                          BEQ     8$              ; IF NOT, BRANCH
     89                                 ;       SEE IF ERROR WAS CAUSE OF ATTENTION
     90 003547  104301  001704                  MOV     ST+ST.ERR,R1    ; GET ERROR BYTE
     91 003551  103201  177407                  BIC     #<HIBYTE!7>,R1  ; CLEAR UNUSED BITS
     92 003553  053560                          BNE     8$              ; IF ERROR(S), BRANCH
     93                                 ;       SEE IF LOGGABLE INFORMATION IS CAUSE OF ATTENTION
     94 003554  102200  000010  001703          BIT     #ST.EL,ST+ST.REQ        ; SEE IF LOGGABLE INFO
     95 003557  013625                          BEQ     50$             ; IF NOT, BRANCH ('RECOVER')
```

```
     1 003560  104650  000063  001106  8$:     MOV     U.UNUM(R5),HRQ.03       ; GET BASE NUMBER
     2 003563  105650  000050  001106          ADD     U.SUBU(R5),HRQ.03       ; ADD SUBUNIT OFFSET
     3 003566  104307  001103                  MOV     HRQ.RQ,R0       ; MOVE REQUEST TO R0
     4 003570  021053                         CALL    HOSTRQ          ; REPORT TO HOST
     5 003571  115000  001104                  TST     HRQ.01          ; SEE IF UNIT DROPPED
     6 003573  073656                         BMI     80$             ; IF SO, BRANCH
     7 003574                                 ASSUME  DROP,100000     ; HOST DROP BIT = BIT 15
     8 003574  104652  000047                  MOV     U.RCOV(R5),R2   ; SEE IF RECOVERY IS ENABLED
     9 003576  073711                         BMI     90$             ; IF NOT, BRANCH
    10 003577                                 ASSUME  ERMASK,100000   ; ASSUME RECOVERY MASK IS SIGN BIT
    11 003577  102200  000002  002226          BIT     #ATTN,ERMODE    ; DID WE HAVE AN ERROR?
    12 003602  013625                         BEQ     50$             ; IF NOT, BRANCH
    13 003603                                 REPSFT  SOFT            ; ELSE, REPORT SOFT ERROR
       003603  104200  000001  001105                                          MOV     #1,HRQ.02
       003606  114000  001106                                                  CLR     HRQ.03
       003610  114000  001107                                                  CLR     HRQ.04
       003612  100467                                                                  MOV R0,-(SP)
       003613  104657  000063                                                  MOV     U.UNUM(R5),R0
       003615  105657  000050                                                  ADD     U.SUBU(R5),R0
       003617  104070  001104                                                  MOV     R0,HRQ.01
       003621  104207  060007                                                  MOV     #T4SOFT,R0
       003623  021053                                                          CALL    HOSTRQ
       003624  104267                                                                  MOV (SP)+,R0
    14 003625  023720          50$:    CALL    RECOVR          ; RECOVER FROM ERRORS
    15 003626  115002                  TST     R2              ; SEE IF ERRORS OCCURRED
    16 003627  053352                  BNE     40$             ; IF SO, LOOP
    17 003630                         POP     <R0,R1>         ; RESTORE R0, R1
       003630  104267                                                          MOV (SP)+,R0
       003631  104261                                                          MOV (SP)+,R1
    18 003632  104653  000047          MOV     U.RCOV(R5),R3   ; GET RECOVERY WORD
    19 003634  104032                  MOV     R3,R2           ; STORE IN R2
    20 003635  103202  001000          BIC     #RETRY,R2       ; CLEAR RETRY BIT
    21 003637  100652  000047          MOV     R2,U.RCOV(R5)   ; STORE RECOVERY WORD
    22 003641  102203  006000          BIT     #SEKREQ!RCBREQ,R3       ; SEE IF SEEK NEEDED
    23 003643  013647                  BEQ     9$              ; IF NOT, BRANCH
    24 003644  104207  000023          MOV     #SEEK,R0        ; SEEK NEXT
    25 003646  003715                  BR      NOSUB           ; EXIT
    26 003647  102203  001000    9$:   BIT     #RETRY,R3       ; SEE IF LAST MODULE SHOULD BE RETRIED
    27 003651  013713                  BEQ     100$            ; IF NOT, BRANCH
    28 003652  104307  002222          MOV     LSTOVL,R0       ; RETRY LAST OVERLAY
    29 003654  114001                  CLR     R1              ; IMMIDATELY
    30 003655  003713                  BR      100$            ; EXIT
    31 003656                  80$:    POP     <R0,R1>         ; RESTORE REGISTERS
       003656  104267                                                          MOV (SP)+,R0
       003657  104261                                                          MOV (SP)+,R1
    32 003660  024212                  CALL    DSABLE          ; DISABLE ERROR RECOVERY
    33 003661  115000  002226          TST     ERMODE          ; SEE IF ENTIRE UNIT DROPPED
    34 003663  033674                  BPL     10$             ; IF NOT, BRANCH
    35 003664                         ASSUME  DROP,100000     ; ASSUME DROP SIGN BIT
    36 003664  114001                  CLR     R1              ; IMMIDATE CALL
    37 003665  104207  000026          MOV     #DRPALL,R0      ; DRPALL NEXT
    38 003667  102200  000001  002223  BIT     #INTINP,M.PARM  ; SEE IF INITIALIZATION IN PROGRESS
    39 003672  013713                  BEQ     100$            ; IF NOT, BRANCH
    40 003673  003715                  BR      NOSUB           ; IF SO, JUST RETURN TO TROOT
    41 003674  104142          10$:    MOV     (R4),R2         ; GET SUBUNIT PARAMETERS
    42 003675                         ASSUME  S.PARM,0
    43 003675  101202  100000          BIS     #DROP,R2        ; DROP SUBUNIT
```

```
     44 003677  100142                          MOV     R2,(R4)           ; SAVE
     45 003700                                  ASSUME  S.PARM,0
     46 003700  104651  000046                  MOV     U.PARM(R5),R1     ; GET UNIT PARAMETERS
     47 003702  101201  001000                  BIS     #FTIME,R1         ; MARK AS FIRST TIME
     48 003704  100651  000046                  MOV     R1,U.PARM(R5)     ; SAVE
     49 003706  104207  000001                  MOV     #SETUP,R0         ; SETUP NEXT MODULE
     50 003710  003715                          BR      NOSUB             ; EXIT
     51 003711                          90$:    POP     <R0,R1>           ; RESTORE REGISTERS
        003711  104267                                                                      MOV (SP)+,R0
        003712  104261                                                                      MOV (SP)+,R1
     52 003713  115001                  100$:   TST     R1                ; SEE IF IMMIDATE CALL
     53 003714  012601                          BEQ                 IF SO, BRANCH
     54 003715  100657  000013          NOSUB:  MOV     R0,U.NFUN(R5)     ; SAVE NEXT FUNCTION
     55 003717  000000                          RETURN                    ; RETURN TO ROOT
     56
```

```
                                        .SBTTL  RECOVR - RECOVER FROM THE ERROR
003720                                  RECOVR:
                                        ;
                                        ;       SET UP DRIVE AS IT SHOULD BE
                                        ;
003720  020746                          CALL    RTDS            ; SEE IF WE CAN GET VALID STATE
003721  115002                          TST     R2              ; SEE IF WE HAVE VALID STATE
003722  054101                          BNE     10$             ; IF NOT, BRANCH
003723  104203  001612                  MOV     #CR.ONL,R3      ; POINT TO ONLINE COMMAND
003725  021173                          CALL    TALK            ; INITIATE SDI INTERCHANGE
003726  115002                          TST     R2              ; SEE IF ERROR OCCURRED
003727  013734                          BEQ     1$              ; IF NOT, BRANCH
003730                                  CERROR  5,#SER2         ; REPORT SECONDARY ERROR
003730  104200  004670  001110                                          MOV     #SER2,HRQ.05
003733  004101                          BR      10$             ; BRANCH TO ERROR LOOP
003734  104203  001617          1$:     MOV     #CR.GST,R3      ; POINT TO GET STATUS
003736  021173                          CALL    TALK            ; SEND SDI INTERCHANGE
003737  115002                          TST     R2              ; SEE IF AN ERROR OCCURRED
003740  013745                          BEQ     2$              ; BRANCH IF NO ERROR
003741                                  CERROR  5,#SER0         ; REPORT SECONDARY ERROR
003741  104200  004633  001110                                          MOV     #SER0,HRQ.05
003744  004101                          BR      10$             ; LOOP AND TRY AGAIN
003745  104207  177777          2$:     MOV     #-1,R0          ; SET UP FOR COMPLEMENT OF STATUS
003747  103307  001703                  BIC     ST+ST.REQ,R0    ; R0 HAS STATUS
003751  102207  000023                  BIT     #ST.PS!ST.SR!ST.RU,R0   ; SEE IF ANY FATAL STATES EXIST
003753  014024                          BEQ     6$              ; IF NOT, BRANCH
003754                                  DEVFTL  23              ; REPORT SWITCH OUT OR SPINDLE NOT READY
003754  104200  002162  001107                                          MOV     #ER23,HRQ.04
003757  104202  047667                                                  MOV     #23!FTLDEV+4000.,R2
003761  104020  001105                                                  MOV     R2,HRQ.02
003763  104200  003763  001104                                          MOV     #.,HRQ.01
003766  104200  060014  001103                                          MOV     #ERRMC,HRQ.RQ
003771  102207  000001                  BIT     #ST.RU,R0       ; SEE IF RUN/STOP SWITCH OUT
003773  014000                          BEQ     3$              ; IF NOT, BRANCH
003774                                  CERROR  5,#SER42        ; REPORT RUN SWITCH OUT
003774  104200  002203  001110                                          MOV     #SER42,HRQ.05
003777  004012                          BR      5$              ; BRANCH
004000  102207  000020          3$:     BIT     #ST.SR,R0       ; SEE IF SPINDLE DROPPED READY
004002  014007                          BEQ     4$              ; IF NOT, BRANCH
004003                                  CERROR  5,#SER43        ; REPORT SPINDLE NOT READY
004003  104200  002217  001110                                          MOV     #SER43,HRQ.05
004006  004012                          BR      5$              ; BRANCH
004007                          4$:     CERROR  5,#SER44        ; REPORT PORT SWITCH OUT
004007  104200  002234  001110                                          MOV     #SER44,HRQ.05
004012                          5$:     ENDERR
004012  104200  000051  001111                                          MOV     #SER22,HRQ.06
004015  104200  000007  002230                                          MOV     #6+1,ERRPOS     ; SET THE POSITION
004020  101200  100000  002230          BIS     #100000,ERRPOS  ; STATUS VALID
004023  004101                          BR      10$             ; BRANCH
004024  102207  000100          6$:     BIT     #ST.RR,R0       ; SEE IF DRIVE IS TO BE RECALIBRATED
004026  054030                          BNE     7$              ; IF NOT, BRANCH (NOTE COMPLEMENT)
004027  024235                          CALL    GORCLB          ; MARK AS RECALIBRATION NEEDED
004030  104307  001703          7$:     MOV     ST+ST.MOD,R0    ; GET MODE BITS
004032  110707                          SWAB    R0              ; MOVE WRITE PROTECT BUTTON STATUS TO LO BYTE
004033  103207  177417                  BIC     #LBHINB,R0      ; CLEAR UNUSED BITS
004035  100657  000026                  MOV     R0,U.WRIT(R5)   ; SAVE WRITE PROTECT STATUS FOR CHANGE MODE
004037  104301  001704                  MOV     ST+ST.ERR,R1    ; GET ERROR BITS
```

```
     46 004041  103201  177400                BIC     #HIBYTE,R1      ; CLEAR UNUSED BITS
     47 004043  014063                      BEQ     9$              ; IF NO ERRORS, BRANCH
     48 004044  102201  000200              BIT     #ST.DE,R1       ; SEE IF A DRIVE ERROR OCCURRED
     49 004046  014050                      BEQ     11$             ; IF NOT, BRANCH
     50 004047  024227                      CALL    GOSEEK          ; SEEK IS REQUIRED
     51 004050  104010  001667      11$:    MOV     R1,DCLR         ; MOVE ERROR BITS TO CLEAR DRIVE COMMAND
     52 004052  104203  001624              MOV     #CR.CLR,R3      ; POINT TO SDI COMMAND
     53 004054  021173                      CALL    TALK            ; INITIATE SDI INTERCHANGE
     54 004055  115002                      TST     R2              ; SEE IF ANY ERRORS OCCURRED
     55 004056  014063                      BEQ     9$              ; IF NOT, BRANCH
     56 004057                              CERROR  5,#SER1         ; REPORT SECONDARY ERROR
        004057  104200  004651  001110                                      MOV     #SER1,HRQ.05
     57 004062  004101                      BR      10$             ; BRANCH TO ERROR LOOP
     58 004063  104653  000026      9$:     MOV     U.WRIT(R5),R3   ; GET WRITE PROTECTION BUTTON STATUS
     59 004065  101653  000045              BIS     U.WPRT(R5),R3   ; SET WRITE PROT BITS FOR READ ONLY DRIVES
     60 004067  101203  177402              BIS     #177402,R3      ; SET OTHER CHANGE MODE BITS
     61 004071  104030  001665              MOV     R3,WRITBT       ; MOVE TO CHANGE MODE COMMAND
     62 004073  104203  001631              MOV     #CR.MOD,R3      ; POINT TO MODE COMMAND
     63 004075  021173                      CALL    TALK            ; INITIATE SDI INTERCHANGE
     64 004076                              CERROR  5,#SER3         ; REPORT SECONDARY ERROR (IF NO ERROR, DON'T CARE)
        004076  104200  004712  001110                                      MOV     #SER3,HRQ.05
     65 004101  114000  002226      10$:    CLR     ERMODE          ; RESET ERMODE
     66 004103  000000                      RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  CHKDN  - FIND PREVIOUS ACTIVE SUBUNIT
004104                                  CHKDN:
                                        ;
                                        ;       SEARCH THE SUBUNIT LIST DOWN AND TRY TO FIND AN ACTIVE SUBUNIT
                                        ;
004104  104203  000001                          MOV     #U.SUBP,R3      ; R3 WILLPOINT TO SUBUNIT POINTERS
004106  105053                                  ADD     R5,R3           ; R3 POINTS TO SUBUNIT POINTERS
004107  117402                          1$:     DEC     R2              ; DECREMENT OLD SUBUNIT
004110  074124                                  BMI     2$              ; IF ALL SUBUNITS CHECKED, BRANCH
004111  104034                                  MOV     R3,R4           ; MOVE SUBUNIT LIST POINTER TO R4
004112  105024                                  ADD     R2,R4           ; POINT TO SUBUNIT
004113  104144                                  MOV     (R4),R4         ; R4 POINTS TO SUBUNIT
004114  074107                                  BMI     1$              ; IF NO UNIT, BRANCH
004115  104147                                  MOV     (R4),R0         ; R0 HAS SUBUNIT PARAMETERS
004116                                          ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
004116  074107                                  BMI     1$              ; IF THIS SUBUNIT DROPPED, BRANCH
004117                                          ASSUME  DROP,100000     ; ASSUME DROP IS SIGN BIT
004117  100652  000050                          MOV     R2,U.SUBU(R5)   ; SAVE SUBUNIT NUMBER
004121  114007                                  CLR     R0              ; FOUND A SUBUNIT
004122  114003                                  CLR     R3              ; PROCESS THIS SUBUNIT
004123  004133                                  BR      3$              ; EXIT
004124  104657  000046                  2$:     MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
004126  103207  010000                          BIC     #DIREC,R0       ; SET DIRECTION UP
004130  100657  000046                          MOV     R0,U.PARM(R5)   ; SAVE
004132  104057                                  MOV     R5,R0           ; DIDN'T FIND A SUBUNIT
004133  000000                          3$:     RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  CHKDN  - FIND FOLLOWING ACTIVE SUBUNIT
004134                                  CHKUP:
                                        ;
                                        ;       SEARCH THE SUBUNIT LIST UP AND TRY TO FIND AN ACTIVE SUBUNIT
                                        ;
004134  104203  000001                          MOV     #U.SUBP,R3      ; R3 WILLPOINT TO SUBUNIT POINTEPS
004136  105053                                 ADD     R5,R3           ; R3 POINTS TO SUBUNIT POINTERS
004137  115402                          1$:     INC     R2              ; INCREMENT OLD SUBUNIT
004140  106202  000003                          CMP     #3,R2           ; SEE IF ALL SUBUNITS CHECKED
004142  074164                                 BMI     3$              ; IF ALL SUBUNITS CHECKED, BRANCH
004143  104034                                 MOV     R3,R4           ; MOVE SUBUNIT LIST POINTER TO R4
004144  105024                                 ADD     R2,R4           ; POINT TO SUBUNIT
004145  104144                                 MOV     (R4),R4         ; R4 POINTS TO SUBUNIT
004146  074137                                 BMI     1$              ; IF NO UNIT, BRANCH
004147  104147                                 MOV     (R4),R0         ; R0 HAS SUBUNIT PARAMETERS
004150                                         ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
004150                                         ASSUME  DROP,100000     ; ASSUME DROP IS SIGN BIT
004150  074137                                 BMI     1$              ; IF THIS SUBUNIT DROPPED, BRANC
004151  115000  002223                         TST     M.PARM          ; SEE IF INITIAL WRITE IN PROGRESS
004153  034157                                 BPL     2$              ; IF NOT, BRANCH
004154                                         ASSUME  IWIPRG,100000   ; ASSUME IWIPRG IS SIGN BIT
004154  102207  040000                         BIT     #INITW,R0       ; SEE IF INITIAL WRITE TO BE DONE ON SUBUNIT
004156  014137                                 BEQ     1$              ; IF NOT, BRANCH
004157  100652  000050          2$:     MOV     R2,U.SUBU(R5)   ; SAVE SUBUNIT NUMBER
004161  114007                                 CLR     R0              ; FOUND ONE
004162  114003                                 CLR     R3              ; PROCESS THIS ONE
004163  004202                                 BR      5$              ; EXIT
004164  104651  000046          3$:     MOV     U.PARM(R5),R1   ; GET UNIT PARAMETERS
004166  115000  002223                         TST     M.PARM          ; SEE IF DOING AN INITIAL WRITE
004170  034176                                 BPL     4$              ; IF NOT, BRANCH
004171                                         ASSUME  IWIPRG,100000   ; ASSUME IWIPRG IS SIGN BIT
004171  103201  040000                         BIC     #INITW,R1       ; FLAG INITIAL WRITE AS COMPLETE ON THIS UNIT
004173  101201  001000                         BIS     #FTIME,R1       ; START TESTING UNIT
004175  004200                                 BR      9$              ; EXIT
004176  101201  010000          4$:     BIS     #DIREC,R1       ; SET DIRECTON DOWN
004200  100651  000046          9$:     MOV     R1,U.PARM(R5)   ; SAVE
004202  000000                  5$:     RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  ERROR RECOVERY SUPPORT SUBROUTINES
                                        .SBTTL  ENABLE - ENABLE ERROR RECOVERY
004203                                  ENABLE:
                                        ;
                                        ;       ENABLE ERROR RECOVERY
                                        ;
004203  104653  000047                  MOV     U.RCOV(R5),R3   ; GET RECOVERY WORD
004205  103203  100000                  BIC     #ERMASK,R3      ; CLEAR MASK
004207  100653  000047                  MOV     R3,U.RCOV(R5)   ; SAVE
004211  000000                          RETURN                  ; RETURN
                                        .SBTTL  DSABLE - DISABLE ERROR RECOVERY
004212                                  DSABLE:
                                        ;
                                        ;       DISABLE ERROR RECOVERY
                                        ;
004212  104653  000047                  MOV     U.RCOV(R5),R3   ; GET RECOVERY WORD
004214  101203  100000                  BIS     #ERMASK,R3      ; SET MASK
004216  100653  000047                  MOV     R3,U.RCOV(R5)   ; SAVE
004220  000000                          RETURN                  ; RETURN
                                        .SBTTL  GORTRY - RETRY OF MODULE REQUIRED TO RECOVER
004221                                  GORTRY:
                                        ;
                                        ;       JUST RETRY THE LAST MODULE TO RECOVER
                                        ;
004221  024203                          CALL    ENABLE          ; ENABLE ERROR RECOVERY
004222  101203  001000                  BIS     #RETRY,R3       ; SET RETRY BIT
004224  100653  000047                  MOV     R3,U.RCOV(R5)   ; SAVE
004226  000000                          RETURN                  ; RETURN
                                        .SBTTL  GOSEEK - SEEK REQUIRED TO RECOVER
004227                                  GOSEEK:
                                        ;
                                        ;       GOSEEK WILL FLAG THAT THE DRIVE MUST SEEK TO RECOVER FROM AN ERROR
                                        ;
004227  024203                          CALL    ENABLE          ; ENABLE ERROR RECOVERY
004230  101203  004000                  BIS     #SEKREQ,R3      ; SET SEEK REQUIRED BIT
004232  100653  000047                  MOV     R3,U.RCOV(R5)   ; SAVE
004234  000000                          RETURN                  ; RETURN
                                        .SBTTL  GORCLB - RECALIBRATE REQUIRED TO RECOVER
004235                                  GORCLB:
                                        ;
                                        ;       GORCLB WILL FLAG THAT THE DRIVE MUST RECALIBRATE TO RECOVER FROM AN ERROR
                                        ;
004235  024203                          CALL    ENABLE          ; ENABLE ERROR RECOVERY
004236  101203  006000                  BIS     #SEKREQ!RCBREQ,R3       ; SET SEEK AND RECALIBRATE REQUIRED BIT
004240  100653  000047                  MOV     R3,U.RCOV(R5)   ; SAVE
004242  000000                          RETURN                  ; RETURN
                                        .SBTTL  GOINIT - DRIVE MUST BE INITIALIZED TO RECOVER
004243                                  GOINIT:
                                        ;
                                        ;       INIT THE DRIVE, THEN FLAG AS DONE
                                        ;
004243  104652  000025                  MOV     U.MASK(R5),R2   ; GET PORT MASK
004245  060011                          XFC     DINIT           ; INIT THE DRIVE
004246  024227                          CALL    GOSEEK          ; SEEK REQUIRED IF INITED
004247  101203  040000                  BIS     #DRINIT,R3      ; MARK THE DRIVE AS INITIALIZED
004251  100653  000047                  MOV     R3,U.RCOV(R5)   ; SAVE
004253  000000                          RETURN                  ; RETURN
```

```
                                .SBTTL  OVERLAY DATA STRUCTURES AND ASSEMBLY ERROR CHECKING
                                ;
                                ;       THE PARTITION TABLE HAS THE PARTITION START ADDRESSES AS WELL
                                ;       AS HOLDING THE NUMBER OF THE OVERLAY LOADED CURRENTLY (IF NO
                                ;       OVERLAY IS LOADED, THE OVERLAY NUMBER IS -1
                                ;
004254                          PTABLE:
004254  177777                  PART0:  .WORD   -1
004255  004422                          .WORD   AREA0
004256  177777                  PART1:  .WORD   -1
004257  004611                          .WORD   AREA1
004260  177777                  PART2:  .WORD   -1
004261  005052                          .WORD   AREA2
004262  177777                  PART3:  .WORD   -1
004263  005237                          .WORD   BUFARA          ; PARTITION 3 GOES IN READ/WRITE BUFFER AREA
004264  177777                  PARTI:  .WORD   -1
004265  004710                          .WORD   AREAI
                                ;
                                ;       EACH ENTRY IN THE OVERLAY TABLE IS COMPOSED OF FOUR WORDS:
                                ;
                                ;       0)      PARTITION TO LOAD OVERLAY INTO
                                ;       1)      NUMBER OF FOLLOWING PARTITIONS THAT ARE DISTROYED
                                ;       2)      LOW ORDER STARTING UNIBUS ADDRESS OF OVERLAY
                                ;       3)      <1::0> HI ORDER STARTING UNIBUS ADDRES
                                ;               <15::2> LENGTH OF OVERLAY
                                ;
        000000                  MSSG$   =       0
        000000                  OCNT$   =       0
004266                          OTABLE: DFOVLY  MSSG$,MS,PART0          ; MESSAGES
004266  000000                          .WORD   <PART0-PART0>/2
004267  000000                          .WORD   0
004270  000000                          .WORD   0
004271  027610                          .WORD   OVL.MS*4
004272                                  DFOVLY  SETUP,SU,PART0,PART3    ; SETUP
004272  000000                          .WORD   <PART0-PART0>/2
004273  000003                          .WORD   <PART3-PART0>/2
004274  000000                          .WORD   0
004275  000734                          .WORD   OVL.SU*4
004276                                  DFOVLY  NEWOP,NO,PART1          ; NEWOP
004276  000001                          .WORD   <PART1-PART0>/2
004277  000000                          .WORD   0
004300  000000                          .WORD   0
004301  001204                          .WORD   OVL.NO*4
004302                                  DFOVLY  RNDBE,RB,PART2          ; RNDBE
004302  000002                          .WORD   <PART2-PART0>/2
004303  000000                          .WORD   0
004304  000000                          .WORD   0
004305  000700                          .WORD   OVL.RB*4
004306                                  DFOVLY          ,PART2          ;       004306          000002
004307  000000                          .WORD   0
004310  000000                          .WORD   0
004311  002574                          .WORD   OVL.SB*4
004312                                  DFOVLY  RNDTG,RT,PART2          ; RNDTG
004312  000002                          .WORD   <PART2-PART0>/2
004313  000000                          .WORD   0
```

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE 44-1
OVERLAY DATA STRUCTURES AND ASSEMBLY ERROR CHECKING

```
        004314  000000                  .WORD   0
        004315  000660                  .WORD   OVL.RT*4
     36 004316                          DFOVLY          ,PART2          ;       004316          000002
        004317  000000                  .WORD   0
        004320  000000                  .WORD   0
        004321  002344                  .WORD   OVL.ST*4
     37 004322                          DFOVLY  BUILDP,BP,PART0,PART3   ; BUILDP
        004322  000000                  .WORD   <PART0-PART0>/2
        004323  000003                  .WORD   <PART3-PART0>/2
        004324  000000                  .WORD   0
        004325  001344                  .WORD   OVL.BP*4
     38 004326                          DFOVLY  WRITE,W,PART0           ; WRITE
        004326  000000                  .WORD   <PART0-PART0>/2
        004327  000000                  .WORD   0
        004330  000000                  .WORD   0
        004331  002020                  .WORD   OVL.W*4
     39 004332                          DFOVLY  AFTWRT,AW,PART0         ; AFTWRT
        004332  000000                  .WORD   <PART0-PART0>/2
        004333  000000                  .WORD   0
        004334  000000                  .WORD   0
        004335  003020                  .WORD   OVL.AW*4
     40 004336                          DFOVLY  READ,R,PART0            ; READ
        004336  000000                  .WORD   <PART0-PART0>/2
        004337  000000                  .WORD   0
        004340  000000                  .WORD   0
        004341  001330                  .WORD   OVL.R*4
     41 004342                          DFOVLY  SECCHK,SC,PART0         ; SECCHK
        004342  000000                  .WORD   <PART0-PART0>/2
        004343  000000                  .WORD   0
        004344  000000                  .WORD   0
        004345  003064                  .WORD   OVL.SC*4
     42 004346                          DFOVLY  CHKEDC,ED,PART0         ; CHKEDC
        004346  000000                  .WORD   <PART0-PART0>/2
        004347  000000                  .WORD   0
        004350  000000                  .WORD   0
        004351  001734                  .WORD   OVL.ED*4
     43 004352                          DFOVLY  CHKECC,CK,PART0         ; CHKECC
        004352  000000                  .WORD   <PART0-PART0>/2
        004353  000000                  .WORD   0
        004354  000000                  .WORD   0
        004355  002074                  .WORD   OVL.CK*4
     44 004356                          DFOVLY  ERCOV,EC,PART0          ; ERCOV
        004356  000000                  .WORD   <PART0-PART0>/2
        004357  000000                  .WORD   0
        004360  000000                  .WORD   0
        004361  000360                  .WORD   OVL.EC*4
     45 004362                          DFOVLY  NEWLEV,NL,PART0         ; NEWLEV
        004362  000000                  .WORD   <PART0-PART0>/2
        004363  000000                  .WORD   0
        004364  000000                  .WORD   0
        004365  000704                  .WORD   OVL.NL*4
     46 004366                          DFOVLY  CMPDAT,CD,PART0         ; CMPDAT
        004366  000000                  .WORD   <PART0-PART0>/2
        004367  000000                  .WORD   0
        004370  000000                  .WORD   0
        004371  001404                  .WORD   OVL.CD*4
```

```
     47 004372                                  DFOVLY  LSTMOD,LM,PART0          ; LSTMOD
        004372  000000                          .WORD   <PART0-PART0>/2
        004373  000000                          .WORD   0
        004374  000000                          .WORD   0
        004375  000270                          .WORD   OVL.LM*4
     48 004376                                  DFOVLY  REVCT,RV,PART0           ; REVCT
        004376  000000                          .WORD   <PART0-PART0>/2
        004377  000000                          .WORD   0
        004400  000000                          .WORD   0
        004401  002640                          .WORD   OVL.RV*4
     49 004402                                  DFOVLY  SEEK,SK,PART3            ; SEEK
        004402  000003                          .WORD   <PART3-PART0>/2
        004403  000000                          .WORD   0
        004404  000000                          .WORD   0
        004405  001514                          .WORD   OVL.SK*4
     50 004406                                  DFOVLY  SEKTST,TS,PART3          ; SEKTST
        004406  000003                          .WORD   <PART3-PART0>/2
        004407  000000                          .WORD   0
        004410  000000                          .WORD   0
        004411  001744                          .WORD   OVL.TS*4
     51 004412                                  DFOVLY  RECALB,RC,PART3          ; RECALB
        004412  000003                          .WORD   <PART3-PART0>/2
        004413  000000                          .WORD   0
        004414  000000                          .WORD   0
        004415  000300                          .WORD   OVL.RC*4
     52 004416                                  DFOVLY  DRPALL,DA,PART0,PART3    ; DRPALL
        004416  000000                          .WORD   <PART0-PART0>/2
        004417  000003                          .WORD   <PART3-PART0>/2
        004420  000000                          .WORD   0
        004421  000254                          .WORD   OVL.DA*4
     53                                 ;
     54                                 ;       EVERYTHING FOLLOWING THE AREA0 DEFINITION IS IN OVERLAY PARTITION
     55                                 ;       SPACE AND WILL BE DESTROYED AFTER INITIALIZATION
     56                                 ;
     57         004422                  AREA0   =       .
     58 004422                                  DFOVLY  INSET,IN,PART1           ; INSET
        004422  000004                          .WORD   <PART1-PART0>/2
        004423  000000                          .WORD   0
        004424  000000                          .WORD   0
        004425  000110                          .WORD   OVL.IN*4
     59 004426                                  DFOVLY  COMCHR,CC,PART1          ; COMCHR
        004426  000004                          .WORD   <PART1-PART0>/2
        004427  000000                          .WORD   0
        004430  000000                          .WORD   0
        004431  000614                          .WORD   OVL.CC*4
     60 004432                                  DFOVLY  SPINUP,SP,PART1          ; SPINUP
        004432  000004                          .WORD   <PART1-PART0>/2
        004433  000000                          .WORD   0
        004434  000000                          .WORD   0
        004435  000134                          .WORD   OVL.SP*4
     61 004436                                  DFOVLY  SORT,SO,PART1            ; SORT
        004436  000004                          .WORD   <PART1-PART0>/2
        004437  000000                          .WORD   0
        004440  000000                          .WORD   0
        004441  001110                          .WORD   OVL.SO*4
     62 004442                                  DFOVLY  SCHAR0,SO,PART1          ; SCHAR0
        004442  000004                          .WORD   <PART1-PART0>/2
```

```
                004443  000000                          .WORD   0
                004444  000000                          .WORD   0
                004445  004404                          .WORD   OVL.S0*4
             63 004446                                  DFOVLY  SCHAR1,S1,PARTI         ; SCHAR1
                004446  000004                          .WORD   <PARTI-PART0>/2
                004447  000000                          .WORD   0
                004450  000000                          .WORD   0
                004451  003120                          .WORD   OVL.S1*4
             64 004452                                  DFOVLY  GETSER,GS,PARTI         ; GETSER
                004452  000004                          .WORD   <PARTI-PART0>/2
                004453  000000                          .WORD   0
                004454  000000                          .WORD   0
                004455  002170                          .WORD   OVL.GS*4
             65 004456                                  DFOVLY  INITD,ID,PARTI          ; INITD
                004456  000004                          .WORD   <PARTI-PART0>/2
                004457  000000                          .WORD   0
                004460  000000                          .WORD   0
                004461  000604                          .WORD   OVL.ID*4
             66         000037              NUMOVL      =       <. - OTABLE> / 4
             67                             .SBTTL      GET CHARACTERISTICS AND SUBUNIT CHARACTERISTICS SDI COMMANDS
             68                             ;
             69                             ;           NOTE: BECAUSE OF THEIR POSITION, THE FOLLOWING SDI LEVEL 2 COMMANDS
             70                             ;           WILL BE ISSUED WITH A LONG TIMEOUT
             71                             ;
             72         000000              SDI$        =       0                   ; DOUBLE UP ON THESE COMMANDS, SINCE IF
             73                                                                     ; AN ERROR OCCURS, IT'S A DEVICE FATAL
             74 004462                      CR.GCR:     MSG GCR,1,11.,CHRRES        ; GET CHARACTERISTICS
                004462  004502                          .WORD   GCR                         ; ADDRESS OF COMMAND
                004463  000001                          .WORD   1                           ; SIZE OF COMMAND IN BYTES
                004464  000013                          .WORD   11.                         ; SIZE OF REPLY IN WORDS
                004465  000170                          .WORD   CHRRES                      ; SUCCESSFUL COMPLETION CODE
                004466  000035                          .WORD   SDI$+U.SDI2                 ; RETRY COUNT OFFSET
             75 004467                      CR.SCR:     MSG SCR,2,19.,SBCRES        ; GET SUBUNIT CHARACTERISTICS
                004467  004503                          .WORD   SCR                         ; ADDRESS OF COMMAND
                004470  000002                          .WORD   2                           ; SIZE OF COMMAND IN BYTES
                004471  000023                          .WORD   19.                         ; SIZE OF REPLY IN WORDS
                004472  000167                          .WORD   SBCRES                      ; SUCCESSFUL COMPLETION CODE
                004473  000036                          .WORD   SDI$+U.SDI2                 ; RETRY COUNT OFFSET
             76 004474                      CR.RUN:     MSG RUN,1,7,COMPLT          ; INITIATE LOAD
                004474  004501                          .WORD   RUN                         ; ADDRESS OF COMMAND
                004475  000001                          .WORD   1                           ; SIZE OF COMMAND IN BYTES
                004476  000007                          .WORD   7                           ; SIZE OF REPLY IN WORDS
                004477  000176                          .WORD   COMPLT                      ; SUCCESSFUL COMPLETION CODE
                004500  000037                          .WORD   SDI$+U.SDI2                 ; RETRY COUNT OFFSET
             84 004501     000      014      RUN:        .BYTE   0,DRVRUN            ; DRIVE RUN
             85 004502     000      207      GCR:        .BYTE   0,GETCHR            ; GET CHARACTERISTICS WITH A
             86 004503     000      210      SCR:        .BYTE   0,GETSUB            ; GET SUBUNIT CHARACTERISTICS
             87 004504  000000               SUBUNT:     .WORD   0                   ; SUBUNIT SELECTION IN LOW ORDER BYTE
             88                             ;
             89                             ;           CHECK THAT 4 UNITS WITH 8 SUBUNITS IN ALL, WITH 16 BAD BLOCKS
             90                             ;           SPECIFIED EACH, CAN FIT IN MEMORY WITHOUT OVERLAPPING WITH
             91                             ;           THE OVERLAYS OR THE R/W BUFFER SPACE
             92                             ;
             93                                         .IF     DF,MAXADR
            101         000010              MAXSUB      =       8.
            105         006232              MINADR      =       HIMEM-<<4*<U.LGRP+1>>+<MAXSUB*<S.BESS+68.>>+2>
            114                                        .ENDC
```

```
                                    .IF     NDF,BUFARA
                            BUFARA  =       0                       ; TO DETERMINE READ/WRITE BUFFERS START
                            MAXADR  =       0                       ; TO DETERMINE MAXIMUM OVERLAY ADDRESS
                                    .ENDC
```

```
                                .SBTTL  ********** NON-OVERLAY MODULE START - TEST 4 INITIALIZATION
                                ;*****************************************************************************
                                ;*****************************************************************************
                                ;*****************************************************************************
                                ;*****************************************************************************
                                ;
                                ;
                                ;       THIS 'OVERLAY' IS LOADED ALONG WITH THE INITIAL DOWNLINE LOAD OF
                                ;       TEST 4.  ONCE EXECUTED, THIS OVERLAY IS NEVER LOADED AGAIN UNLESS
                                ;       THE ENTIRE TEST IS STARTED (OR RESTARTED) AGAIN.  ALL OTHER
                                ;       OVERLAYS ARE LOADED INTO THE SPACE THAT THIS ONE OCCUPIES INITIALLY,
                                ;       THUS DESTROYING THIS OVERLAY.
                                ;
                                ;
004505                          UCURSR: .BLKW   1
004506  007702                  LASTU:  .WORD   FIRSTU
004507                          UMASK:  .BLKW   1
004510                          NUMBB:  .BLKW   1
004511                          MAXDBN: .BLKW   2
004513                          SECTRK: .BLKW   1
004514                          SECGRP: .BLKW   1
004515                          SECCYL: .BLKW   1
004516                          TGS:    .BLKW   7
004525                          DSERNM: .BLKW   3               ; TEMP STORAGE FOR DRIVE SERIAL NUMBER
```

********** NON-OVERLAY MODULE START - TEST 4 INITIALIZATION

```
      1 004530  024710                    START:  CALL    GMPARM          ; GET MASTER PARAMETERS
      2 004531  024764                            CALL    GETU            ; GET ALL UNITS TO TEST
      3 004532  025374                            CALL    BLDUNT          ; BUILD UNIT AND SUBUNIT PARAMETERS
      4 004533  101200  000001  002223            BIS     #INTINP,M.PARM  ; MARK AS INITIALIZATION IN PROGRESS
      5 004536  024647                            CALL    TROOT           ; FILL OUT UNIT AND SUBUNIT PARM'S, GET SUBUNIT CHAR
      6 004537  102200  000002  002223            BIT     #DIE,M.PARM     ; SEE IF INITIALIZATION ERRORS
      7 004542  014572                            BEQ     3$              ; IF NOT, BRANCH
      8 004543                                    DEVFTL  16.,#SER39      ; REPORT INITIALIZATION ERRORS
        004543  104200  001556  001107                                            MOV     #ER16,HRQ.04
        004546  104200  002502  001110                                            MOV     #SER39,HRQ.05
        004551  104202  047660                                                    MOV     #16!FTLDEV+4000.,R2
        004553  104020  001105                                                    MOV     R2,HRQ.02
        004555  104200  004555  001104                                            MOV     #.,HRQ.01
        004560  104200  060014  001103                                            MOV     #ERRMC,HRQ.RQ
      9 004563  104200  177777  001106            MOV     #-1,HRQ.03      ; MOVE 'NOT ASSICIATED WITH ANY UNIT' TO UNIT #
     10 004566  104307  001103                    MOV     HRQ.RQ,R0       ; GET REQUEST
     11 004570  021053                            CALL    HOSTRQ          ; REPORT
     12 004571  002573                            BR      STOP            ; STOP TEST 4
     13 004572  103200  000001  002223  3$:       BIC     #INTINP,M.PARM  ; INITIALIZATION NO LONGER IN PROGRESS
     14 004575  104307  002227                    MOV     MEMPOL,R0       ; R0 HAS POINTER TO FREE MEMORY
     15 004577  107207  005237                    SUB     #BUFARA,R0      ; SUBTRACT END OF CODE FROM FREE MEMORY
     16 004601  104073                            MOV     R0,R3           ; SAVE AMOUNT OF FREE MEMORY
     17 004602  114001                            CLR     R1              ; CLEAR COUNT
     18 004603  107207  000424            1$:     SUB     #RBUFLN+LINKLN,R0       ; SUBTRACT NEEDED MEM FOR READ
     19 004605  074610                            BMI     2$              ; IF NO MORE MEMORY, BRANCH
     20 004606  115401                            INC     R1              ; INCREMENT COUNT
     21 004607  004603                            BR      1$              ; LOOP
     22 004610  104010  002233            2$:     MOV     R1,SECMAX       ; SAVE IN SECTOR MAXIMUM
     23 004612  114001                            CLR     R1              ; CLEAR COUNT
     24 004613  107203  000410                    SUB     #WBUFLN+LINKLN,R3       ; SUBTRACT NEEDED MEM FOR WRITE
     25 004615  115401            4$:     INC     R1              ; INCREMENT COUNT
     26 004616  107203  000007                    SUB     #LINKLN,R3      ; SUBTRACT LINK LENGTH FROM MEMORY
     27 004620  074622                            BMI     5$              ; IF MEMORY EXHAUSTED, BRANCH
     28 004621  004615                            BR      4$              ; LOOP
     29 004622  104010  002234            5$:     MOV     R1,CHNMAX       ; SAVE IN SECTOR MAXIMUM
     30 004624  002532                            BR      ROOT            ; START EXERCISE
```

```
      1                                     .SBTTL  GETMEM - ALLOCATES MEMORY FOR UNIT AND SUBUNIT BLOCK AND SUBUNIT PARAMETERS
      2 004625                              GETMEM:
      3                                     ;
      4                                     ;       RETURN A BLOCK OF MEMORY
      5                                     ;
      6 004625  107070  002227                     SUB     R0,MEMPOL       ; SUBTRACT REQUESTED LENGTH FROM MEMOY POOL
      7 004627  104307  002227                     MOV     MEMPOL,R0       ; LOAD R0 WITH POINTER TO REQUESTED MEMORY
     22 004631  000000                            RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  CBB2   - 28 BIT COMPARE FOR SETUP MODULES
002632                                  CBB2:
                                        ;
                                        ;       28 BIT COMPARE FOR BAD BLOCKS
                                        ;
006632  104634  000001                          MOV     1(R3),R4        ; GET WORD THAT R3 POINTS TO
006634  103204  170000                          BIC     #^CHBHINB,R4    ; STRIP OFF UNUSED BITS
006636  104625  000001                          MOV     1(R2),R5        ; GET OTHER WORD TO COMPARE
006640  103205  170000                          BIC     #^CHBHINB,R5    ; STRIP OFF THE UNUSED BITS
006642  106045                                  CMP     R4,R5           ; COMPARE
006643  054646                                  BNE     1$              ; IF DIFFERENCE IS FOUND, BRANCH
006644  104125                                  MOV     (R2),R5         ; GET OTHER WORD TO COMPARE
006645  106135                                  CMP     (R3),R5         ; COMPARE
006646  000000                          1$:     RETURN                  ; RETURN TO SORTBE
```

```
                                        .SBTTL  TROOT  - TEMPORARY ROOT FOR          URING TEST 4 SETUP
004647                                  TROOT:
                                        ;
                                        ;       TROOT WILL FILL OUT THE AS YET UNDEFINED FIELDS IN BOTH UNIT
                                        ;       AND SUBUNIT PARAMETERS, AS WELL AS GETTING THE SUBUNIT CAHRACTERISTICS
                                        ;
004647  104205  007702                        MOV     #FIRSTU,R5      ; R5 POINTS TO FIRST SUBUNIT
004651  114002                          MKLOOP: CLR     R2              ; FOR ZEROING RECOVERY WORD
004652  100652  000047                        MOV     R2,U.RCOV(R5)   ; ZERO RECOVERY WORD
004654  104207  001000                        MOV     #FTIME,R0       ; MARK AS FIRST TIME (WILL DISABLE RECOVERY)
004656  100657  000046                        MOV     R0,U.PARM(R5)   ; SAVE
004660  104201  000027                        MOV     #INSET,R1       ; FIRST MODULE IS INSET
004662  100651  000013                        MOV     R1,U.NFUN(R5)   ; SAVE IN UNIT PARAMETERS
004664  104201  000003                        MOV     #3,R1           ; SDI SHORT TIMEOUT SET TO 30 SEC
004666  100651  000033                        MOV     R1,U.SDIS(R5)   ; MOVE TO PARAMETERS
004670  104201  000360                        MOV     #360,R1         ; WRITE PROTECT THE ENTIRE DRIVE
004672  100651  000045                        MOV     R1,U.WPRT(R5)   ; SAVE
004674  022577                                CALL            ; RUN       R
004675  104657  000046                        MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
004677  101207  001000                        BIS     #FTIME,R0       ; SET FIRST TIME
004701  100657  000046                        MOV     R0,U.PARM(R5)   ; SAVE
004703  104155                                MOV     (R5),R5         ; TRAVERSE TO NEXT UNIT
004704                                        ASSUME  U.NEXT,0
004704  106205  007702                        CMP     #FIRSTU,R5      ; SEE IF TRAVERSED ENTIRE RING
004706  054651                                BNE     MKLOOP          ; IF NOT, BRANCH
004707  000000                                RETURN                  ; RETURN TO CALLING PROGRAM
        004710                          AREAI   =       .               ; INITILIZATION OVERLAY AREA
```

```
                                        .SBTTL  GMPARM - GET MASTER PARAMETERS (PATTERN 16) AND SET UP OVERLAY ADDRESSES
004710                                  GMPARM:
                                        ;
                                        ;       GET MASTER PARAMETERS
                                        ;
004710  104302  007774                  MOV     OVSTRT,R2       ; GET STARTING ADDRESS OF OVERLAY (LO)
004712  104303  007775                  MOV     OVSTRT+1,R3     ; GET STARTING ADDRESS OF OVERLAY (HI)
004714  104020  004270                  MOV     R2,OTABLE+2     ; MOVE STARTING ADDRESS INTO OVERLAY TABLE
004716  104207  000036                  MOV     #NUMOVL-1,R0    ; R0 HAS NUMBER OF OVERLAYS
004720  104205  004266                  MOV     #OTABLE,R5      ; R5 POINTS TO OVERLAY TABLE
004722  104651  000003          MORE:   MOV     3(R5),R1        ; R1 HAS OVERLAY LENGTH
004724  110601                          ROR     R1              ; SHIFT TO MAKE BYTES
004725  103201  100001                  BIC     #100001,R1      ; CLEAR UNUSED BITS
004727  105012                          ADD     R1,R2           ; FIND ADDRESS OF NEXT OVERLAY
004730  044732                          BCC     1$              ; IF NO CARRY, BRANCH
004731  115403                          INC     R3              ; PROPOGATE CARRY
004732  100652  000006          1$:     MOV     R2,6(R5)        ; STORE STARTING ADDRESS OF NEXT OVERLAY
004734  104651  000007                  MOV     7(R5),R1        ; GET OVERLAY LENGTH (<<2)
004736  101031                          BIS     R3,R1           ; SET HI ORDER OVERLAY ADDRESS
004737  100651  000007                  MOV     R1,7(R5)        ; SAVE
004741  105205  000004                  ADD     #4,R5           ; POINT TO NEXT OVERLAY AREA
004743  117407                          DEC     R0              ; DECREMENT COUNT
004744  054722                          BNE     MORE            ; IF ALL OVERLAYS NOT SET UP, BRANCH
004745  104207  060003                  MOV     #T4MPRM,R0      ; R0 CONTAINS HOST REQUEST
004747  021053                          CALL    HOSTRQ          ; INITIATE HOST REQUEST
004750  104207  001104                  MOV     #HRQ.01,R0      ; R0 POINTS TO PATTERN INFORMATION
004752  104272                          MOV     (R0)+,R2        ; R2 HAS LENGTH OF PATTERN
004753  014763                          BEQ     3$              ; IF NO PATTERN, BRANCH
004754  104201  002257                  MOV     #PAT0+1,R1      ; R1 POINTS TO PATTERN 16 AREA (SKIP EDC)
004756  100212                          MOV     R2,(R1)+        ; MOVE PATTERN LENGTH TO PAT0 AREA
004757  104273                  2$:     MOV     (R0)+,R3        ; GET WORD OF PATTERN
004760  100213                          MOV     R3,(R1)+        ; MOVE TO PATTERN 16 AREA
004761  117402                          DEC     R2              ; DECREMENT COUNT
004762  054757                          BNE     2$              ; IF COUNT UNEXHAUSTED, BRANCH
004763  000000                  3$:     RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  GETU  - POLL ALL PORTS, THEN GET UNITS TO TEST
004764                                  GETU:
                                        ;
                                        ;       GET THE UNITS TO TEST
                                        ;
                                        ;
                                        ;       POLL ALL PORTS AND FILL IN A UDA PORT INFORMATION TABLE (UNITS)
                                        ;
004764  104205  000001                  MOV     #1,R5           ; MOVE INITIAL MASK TO R5
004766  104204  005354                  MOV     #UNITS,R4       ; R4 POINTS TO UNIT TABLE
004770                          5$:     PUSH    R4              ; SAVE R4
004770  100464                                                          MOV R4,-(SP)
004771  104052                          MOV     R5,R2           ; MOVE MASK TO R2
004772  021007                          CALL    RDSTAT          ; GET DRIVE'S STATUS
004773  115002                          TST     R2              ; SEE IF ERROR
004774  015006                          BEQ     20$             ; IF NOT, BRANCH
004775  075001                          BMI     10$             ; IF MSB, PARITY ERRORS > HALF SECOND
004776  104203  005210                  MOV     #SER10,R3       ; NO DRIVE ATTACHED
005000  005003                          BR      15$             ; BRANCH
005001  104203  002246          10$:    MOV     #SER45,R3       ; PARITY ERRORS > HALF SECOND
005003  100643  000001          15$:    MOV     R3,1(R4)        ; SAVE ERROR MESSAGE
005005  005200                          BR      100$            ; REPORT
005006  114003                  20$:    CLR     R3              ; SET UP TIMEOUT COUNT
005007  104052                  25$:    MOV     R5,R2           ; MOVE PORT SELECT TO R2
005010  021007                          CALL    RDSTAT          ; GET STATUS
005011  115002                          TST     R2              ; SEE IF ERROR
005012  055016                          BNE     30$             ; IF SO, BRANCH
005013  102201  000001                  BIT     #RCVRDY,R1      ; SEE IF RECEIVER READY ASSERTED
005015  055025                          BNE     35$             ; IF SO, BRANCH
005016  117403                  30$:    DEC     R3              ; DECREMENT COUNT
005017  055007                          BNE     25$             ; IF INCOMPLETE, BRANCH
005020  104203  005223                  MOV     #SER11,R3       ; RECEIVER READY NEVER ASSERTED
005022  100643  000001                  MOV     R3,1(R4)        ; SAVE ERROR MESSAGE
005024  005200                          BR      100$            ; REPORT
005025  102201  000100          35$:    BIT     #AVAIL,R1       ; SEE IF DRIVE IS AVAILABLE
005027  055035                          BNE     40$             ; IF SO, BRANCH
005030  104203  005440                  MOV     #SER40,R3       ; GET SECONDARY ERROR
005032  100643  000001                  MOV     R3,1(R4)        ; SAVE
005034  005200                          BR      100$            ; EXIT
005035  104202  001750          40$:    MOV     #MAXSND,R2      ; SET UP MAXIMUM TRIES AT SENDING
005037  104203  001617                  MOV     #CR.GST,R3      ; R3 POINTS TO GET STATUS COMMAND
005041  104137                  45$:    MOV     (R3),R0         ; SET ADR OF SDI COMMAND BUFFER
005042                                  ASSUME  L2.OPC,0
005042  104631  000001                  MOV     L2.SLN(R3),R1   ; SET BUFFER LENGTH
005044                                  PUSH    R2              ; SAVE R2
005044  100462                                                          MOV R2,-(SP)
005045  104052                          MOV     R5,R2           ; SETUP FOR SEND
005046  060004                          XFC     SEND            ; SEND COMMAND
005047                                  POP     R2              ; RESTORE COUNT
005047  104262                                                          MOV (SP)+,R2
005050  115001                          TST     R1              ; DID UNIT ACCEPT COMMAND
005051  015061                          BEQ     50$             ; IF SO, BRANCH
005052  117402                          DEC     R2              ; DECREMENT COUNT
005053  055041                          BNE     45$             ; IF UNEXPIRED, BRANCH
005054  104203  005241                  MOV     #SER12,R3       ; GET ERROR NUMBER
005056  100643  000001                  MOV     R3,1(R4)        ; SAVE
005060  005200                          BR      100$
```

GETU - POLL ALL PORTS, THEN GET UNITS TO TEST

```
     55 005061                          50$:    PUSH    R4              ; SAVE R4
        005061  100464                                                              MOV R4,-(SP)
     56 005062  104204  000003                   MOV     #3,R4           ; SET UP SHORT TIMEOUT
     57 005064  104207  001702          55$:     MOV     #ST,R0          ; SET DATA BUFFER ADDRESS
     58 005066  104631  000002                   MOV     L2.RLN(R3),R1   ; SET BUFFER LENGTH
     59 005070  104052                           MOV     R5,R2           ; SETUP FOR RECEIVE
     60 005071  060005                           XFC     RCV             ; RECEIVE SDI COMMAND
     61 005072  115001                           TST     R1              ; DID ERROR OCCUR
     62 005073  015133                           BEQ     85$             ; IF NOT, BRANCH
     63 005074  106201  000001                   CMP     #1,R1           ; SEE IF TIMEOUT
     64 005076  055105                           BNE     60$             ; IF NOT, BRANCH
     65 005077  117404                           DEC     R4              ; DECREMENT TIMEOUT VALUE
     66 005100  055064                           BNE     55$             ; IF NOT TIMEOUT, BRANCH
     67 005101                                   POP     R4              ; RESTORE R4
        005101  104264                                                              MOV (SP)+,R4
     68 005102  104203  005253                   MOV     #SER13,R3       ; GET ERROR NUMBER
     69 005104  005130                           BR      80$             ; BRANCH TO EXIT
     70 005105                          60$:     POP     R4              ; RESTORE R4
        005105  104264                                                              MOV (SP)+,R4
     71 005106  110601                           ROR     R1              ; ROTATE INTO POSITION TO TEST
     72 005107  110601                           ROR     R1              ; SEE IF FIRST WORD NOT START FRAME
     73 005110  045114                           BCC     65$             ; IF NOT, BRANCH
     74 005111  104203  005266                   MOV     #SER14,R3       ; GET ERROR NUMBER
     75 005113  005130                           BR      80$             ; BRANCH TO END OF LOOP
     76 005114  110601                  65$:     ROR     R1              ; SEE IF FRAMING ERROR
     77 005115  045121                           BCC     70$             ; IF NOT, BRANCH
     78 005116  104203  005314                   MOV     #SER15,R3       ; GET ERROR NUMBER
     79 005120  005130                           BR      80$             ; BRANCH TO END OF LOOP
     80 005121  110601                  70$:     ROR     R1              ; SEE IF CHECKSUM ERROR
     81 005122  045126                           BCC     75$             ; IF NOT, BRANCH
     82 005123  104203  005336                   MOV     #SER16,R3       ; GET ERROR NUMBER
     83 005125  005130                           BR      80$             ; BRANCH TO END OF LOOP
     84 005126  104203  005361          75$:     MOV     #SER17,R3       ; GET ERROR NUMBER
     85 005130  100643  000001          80$:     MOV     R3,1(R4)        ; SAVE
     86 005132  005200                           BR      100$            ; BRANCH TO END OF LOOP
```

```
                                        ;
                                        ;       NOW FILL IN THE TABLE WITH ALL THE SUBUNIT NUMBERS
                                        ;
005133                          85$:    POP     R4              ; RESTORE R4
005133  104264                                                                 MOV (SP)+,R4
005134  104207  177777                  MOV     #-1,R0          ; GET 'NO UNITS' FLAG
005136  100647  000001                  MOV     R0,1(R4)        ; CLEAR ANY ERRORS THAT ARE FLAGGED
005140  100647  000002                  MOV     R0,2(R4)        ; CLEAR ANY ERRORS THAT ARE FLAGGED
005142  104307  001702                  MOV     ST,R0           ; R0 HAS UNIT NUMBER
005144  104072                          MOV     R0,R2           ; COPY R0 TO R2
005145  103207  170000                  BIC     #^CHBHINB,R0    ; R0 HAS UNIT NUMBER
005147                                  PUSH    <R1,R2>         ; SAVE R1 AND R2
005147  100461                                                                 MOV R1,-(SP)
005150  100462                                                                 MOV R2,-(SP)
005151  104052                          MOV     R5,R2           ; MOVE UDA PORT MASK TO R2
005152  021007                          CALL    RDSTAT          ; GET STATUS
005153  102201  000002                  BIT     #ATTN,R1        ; SEE IF SPINABLE
005155  055160                          BNE     90$             ; IF SO, BRANCH
005156  101207  010000                  BIS     #10000,R0       ; SET 'NOT SPINABLE' FLAG
005160                          90$:    POP     <R2,R1>         ; RESTORE
005160  104262                                                                 MOV (SP)+,R2
005161  104261                                                                 MOV (SP)+,R1
005162  101207  040000                  BIS     #40000,R0       ; FLAG UNIT AS NOT TESTED
005164  110702                          SWAB    R2              ; SWAP R2'S BYTES
005165  110602                          ROR     R2              ; MOVE SUBUNIT MASK TO LO NIBBLE
005166  110602                          ROR     R2              ; MOVE SUBUNIT MASK TO LO NIBBLE
005167  110602                          ROR     R2              ; MOVE SUBUNIT MASK TO LO NIBBLE
005170  110602                          ROR     R2              ; MOVE SUBUNIT MASK TO LO NIBBLE
005171  103202  177760                  BIC     #LBLONB,R2      ; CLEAR ALL BUT SUBUNIT BITS
005173  110602                  95$:    ROR     R2              ; MOVE SUBUNIT BIT TO CARRY
005174  045200                          BCC     100$            ; IF NO MORE SUBUNITS, BRANCH
005175  100247                          MOV     R0,(R4)+        ; MOVE SUBUNIT NUMBER TO TABLE
005176  115407                          INC     R0              ; INCREMENT SUBUNIT NUMBER
005177  005173                          BR      95$             ; BRANCH
                                        ;
                                        ;
                                        ;
005200                          100$:   POP     R4              ; R4 POINTS TO START OF UNIT JUST HANDLED
005200  104264                                                                 MOV (SP)+,R4
005201  105204  000004                  ADD     #4,R4           ; R4 WILL POINT TO NEXT UNIT (CLEAR CARRY FOR ROL)
005203  110205                          ROL     R5              ; R5 HAS NEXT UNIT PORT MASK
005204  106205  000020                  CMP     #20,R5          ; SEE IF ALL PORTS TESTED
005206  054770                          BNE     5$              ; IF NOT, BRANCH
```

```
                                        ;
                                        ;       NOW GET THE PLUG NUMBERS TO TEST AND FIND THEM IN THE TABLE
                                        ;
 4 005207  104207  060012              MOV     #UTOTST,R0      ; GET WHAT SUBUNIT NUMBERS TO TEST REQUEST
 5 005211  021053                      CALL    HOSTRQ          ; GET THE PLUG NUMBERS
 6 005212  104207  001104              MOV     #HRQ.01,R0      ; R0 POINTS TO UNIT NUMBERS TO TEST
 7 005214  104201  005354      105$:   MOV     #UNITS,R1       ; R1 POINTS TO UDA PORT INFORMATION
 8 005216  104112              110$:   MOV     (R1),R2         ; R2 HAS UNIT
 9 005217  075242                      BMI     130$            ; IF NONE, BRANCH
10 005220                              PUSH    R1              ; SAVE R1
   005220  100461                                                                  MOV R1,-(SP)
11 005221  103202  160000      115$:   BIC     #160000,R2      ; CLEAR 'NOT TESTED', LEAVE 'UNSPINABLE' SET
12 005223  106172                      CMP     (R0),R2         ; SEE IF IT IS A UNIT TO TEST
13 005224  055230                      BNE     120$            ; NO MATCH
14 005225  100112                      MOV     R2,(R1)         ; SAVE UNIT AS ONE TO TEST
15 005226                              POP     R1              ; RESTORE STACK
   005226  104261                                                                  MOV (SP)+,R1
16 005227  005344                      BR      175$            ; LOOK FOR THE NEXT ONE
17 005230  115401              120$:   INC     R1              ; POINT TO NEXT SUBUNIT
18 005231  104012                      MOV     R1,R2           ; COPY TO R2
19 005232  107202  005354              SUB     #UNITS,R2       ; SUBTRACT STARTING ADDRESS
20 005234  102202  000003              BIT     #3,R2           ; SEE IF STILL ON SAME UNIT
21 005236  015241                      BEQ     125$            ; IF NOT, BRANCH
22 005237  104112                      MOV     (R1),R2         ; SEE IF ANY MORE SUBUNITS
23 005240  035221                      BPL     115$            ; IF SO, BRANCH
24 005241                      125$:   POP     R1              ; RESTORE R1
   005241  104261                                                                  MOV (SP)+,R1
25 005242  105201  000004      130$:   ADD     #4,R1           ; LOOK AT NEXT UNIT
26 005244  106201  005374              CMP     #UNITS+16.,R1   ; SEE IF ENTIRE TABLE SEARCHED
27 005246  055216                      BNE     110$            ; IF NOT, BRANCH
```

GETU - POLL ALL PORTS, THEN GET UNITS TO TEST

```
                                        ;
                                        ;       DIDN'T FIND THE REQUESTED UNITS -- DUMP ALL KNOWLEDGE OF THE
                                        ;       UDA PORTS AND DIE
                                        ;
005247  104170  001106                  MOV     (R0),HRQ.03     ; SAVE UNIT NUMBER IN REQUEST BUFFER
005251  104204  001110                  MOV     #HRQ.05,R4      ; R4 POINTS TO OUTPUT BUFFER
005253  104205  005354                  MOV     #UNITS,R5       ; R5 POINTS TO UNIT TABLE
005255  104157                  135$:   MOV     (R5),R0         ; GET FIRST WORD OF UNIT
005256  035263                          BPL     140$            ; IF VALID UNIT WAS FOUND, BRANCH
005257  104657  000001                  MOV     1(R5),R0        ; GET POINTER TO ERROR MESSAGE
005261  100247                          MOV     R0,(R4)+        ; SAVE IN OUTPUT BUFFER
005262  005316                          BR      170$            ; EXIT
005263                          140$:   PUSH    R5              ; SAVE POINTER TO UNIT TABLE
005263  100465                                                                  MOV R5,-(SP)
005264  102207  010000                  BIT     #10000,R0       ; SEE IF DRIVE UNSPINABLE
005266  015272                          BEQ     145$            ; IF SPINABLE, BRANCH
005267  104207  005462                  MOV     #SER41,R0       ; REPORT DRIVE(S) UNSPINABLE
005271  005274                          BR      150$            ; BRANCH
005272  104207  005413          145$:   MOV     #SER18,R0       ; GET POINTER TO ERROR MESSAGE
005274  100247                  150$:   MOV     R0,(R4)+        ; SAVE
005275  114007                          CLR     R0              ; CLEAR COUNT
005276  104251                  155$:   MOV     (R5)+,R1        ; GET UNIT NUMBER
005277  075304                          BMI     160$            ; IF INVALID, BRANCH
005300  115407                          INC     R0              ; INCREMENT COUNT
005301  106207  000004                  CMP     #4,R0           ; SEE IF MAX
005303  055276                          BNE     155$            ; IF NOT, LOOP
005304  104671  005347          160$:   MOV     SER18E-1(R0),R1 ; GET POINTER TO CORRECT ERROR MESSAGE
005306  100241                          MOV     R1,(R4)+        ; MOVE INTO OUTPUT BUFFER
005307                                  POP     R5              ; .ESTORE R5
005307  104265                                                                  MOV (SP)+,R5
005310                                  PUSH    R5              ; SAVE R5
005310  100465                                                                  MOV R5,-(SP)
005311  104251                  165$:   MOV     (R5)+,R1        ; GET SUBUNIT NUMBER
005312  100241                          MOV     R1,(R4)+        ; SAVE
005313  117407                          DEC     R0              ; DECREMENT COUNT
005314  055311                          BNE     165$            ; IF COUNT INCOMPLETE, BRANCH
005315                                  POP     R5              ; RESTORE R5
005315  104265                                                                  MOV (SP)+,R5
005316  105205  000004          170$:   ADD     #4,R5           ; POINT TO NEXT UNIT TABLE
005320  106205  005374                  CMP     #UNITS+16.,R5   ; SEE IF ENTIRE TABLE SEARCHED
005322  055255                          BNE     135$            ; IF NOT, BRANCH
005323                                  DEVFTL  1000            ; REPORT FATAL ERROR (WILL SHOW UP AS A 5000)
005323  104200  005071  001107                                          MOV     #ER1000,HRQ.04
005326  104202  051610                                                  MOV     #1000!FTLDEV+4000.,R2
005330  104020  001105                                                  MOV     R2,HRQ.02
005332  104200  005332  001104                                          MOV     #.,HRQ.01
005335  104200  060014  001103                                          MOV     #ERRMC,HRQ.RQ
005340  104307  001103                  MOV     HRQ.RQ,R0       ; SET UP FOR REPORT
005342  021053                          CALL    HOSTRQ          ; REPORT ERROR
005343  002573                          BR      STOP            ; END TEST

005344  115407                  175$:   INC     R0              ; POINT TO NEXT UNIT TO TEST
005345  104172                          MOV     (R0),R2         ; CHECK NEXT UNIT
005346  035214                          BPL     105$            ; FIND IN UDA PORT INFORMATION
005347  000000                          RETURN                  ; RETURN TO INITIALIZATION CODE

005350  005435          SER18E: .WORD   SER18A          ; FOR MULTIPLE SUBUNIT ERROR REPORTING
```

```
UCAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE 54-1
GETU   - POLL ALL PORTS, THEN GET UNITS TO TEST

     49 005351  005431                          .WORD   SER18B
     50 005352  005425                          .WORD   SER18C
     51 005353  005421                          .WORD   SER18D
     52
     53 005354  000020                UNITS:    .REPT   16.
     54                                         .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
     55                                         .ENDR
        005354  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005355  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005356  177777                          .WORD   1777,7      ; UNITS AND THEIR SURPERVISOR NUM
        005357  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005360  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005361  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005362  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005363  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005364  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005365  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005366  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005367  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005370  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005371  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005372  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
        005373  177777                          .WORD   177777      ; UNITS AND THEIR SURPERVISOR NUM
```

```
      1                                          .SBTTL  BLDUNT - BUILD THE UNIT PARAMETER BLOCK
      2 005374                                   BLDUNT:
      3                                          ;
      4                                          ;       BLDUNT WILL BUILD THE UNIT AND SUBUNIT PARAMETERS
      5                                          ;
      6 005_ 4  104202  000001                           MOV     #1,R2           ; PORT 1 MASK
      7 005376  104020  004507                           MOV     R2,UMASK        ; SAVE MASK
      8 005400  104204  005354                           MOV     #UNITS,R4       ; POINT TO UNITS TO BE TESTED TABLE
      9 005402  104147                           ULOP:   MOV     (R4),R0         ; GET UNIT TO BE TESTED
     10 005403  075465                                   BMI     NOU             ; IF FIRST UNIT NOT TO BE TESTED, BRANCH
     11 005404  102207  040000                           BIT     #40000,R0       ; SEE IF FIRST SUBUNIT TO BE TESTED
     12 005406  015426                                   BEQ     3$              ; IF SO, BRANCH
     13 005407  104647  000001                           MOV     1(R4),R0        ; GET SECOND SUBUNIT INFORMATION
     14 005411  102207  040000                           BIT     #40000,R0       ; SEE IF THIS SUBUNIT IS USED
     15 005413  015426                                   BEQ     3$              ; IF SO, BRANCH
     16 005414  104647  000002                           MOV     2(R4),R0        ; SEE IF THIRD SUBUNIT TO BE TESTED
     17 005416  102207  040000                           BIT     #40000,R0       ; CHECK TESTING BIT
     18 005420  015426                                   BEQ     3$              ; IF IT IS TO BE TESTED, BRANCH
     19 005421  104647  000003                           MOV     3(R4),R0        ; SEE IF FOURTH SUBUNIT TO BE TESTED
     20 005423  102207  040000                           BIT     #40000,R0       ; TEST TESTING BIT
     21 005425  055465                                   BNE     NOU             ; IF UNIT (THE WHOLE THING) NOT TO BE TESTED, BRANCH
     22 005426  104207  000072                 3$:       MOV     #TLEN.U,R0      ; R0 HAS UNIT DATA STRUCTURE LENGTH
     23 005430  024625                                   CALL    GETMEM          ; GET MEMORY
     24 005431  100707  177053                           MOV     R0,@LASTU       ; LINK LAST POINTER TO THIS ONE
     25 005433                                          ASSUME  U.NEXT,0
     26 005433  104070  004506                           MOV     R0,LASTU        ; SAVE POINTER TO THIS ONE
     27 005435  100672  000025                           MOV     R2,U.MASK(R0)   ; SAVE PORT SELECT MASK
     28 005437  104203  000004                           MOV     #4,R3           ; MAXIMUM OF FOUR SUBUNITS
     29 005441                                          PUSH    R0              ; SAVE POINTER TO UNIT INFORMATION
        005441  100467                                                                      MOV R0,-(SP)
     30 005442  104145                                   MOV     (R4),R5         ; GET STARTING SUBUNIT NUMBER
     31 005443  103205  170000                           BIC     #^CHBHINB,R5    ; CLEAR 'NOT USED' BIT, IF SE;
     32 005445  100675  000063                           MOV     R5,U.UNUM(R0)   ; SAVE STARTING SUBUNIT NUMBER
     33 005447  105207  000001                           ADD     #U.SUBP,R0      ; R0 WILL POINT TO SUBUNIT POINTERS
     34 005451  104245                           1$:     MOV     (R4)+,R5        ; MOVE SUBUNIT NUMBER TO UNIT PARAMETERS
     35 005452  102205  040000                           BIT     #40000,R5       ; SEE IF SUBUNIT TO BE TESTED
     36 005454  015457                                   BEQ     2$              ; IF SO, BRANCH
     37 005455  104205  177777                           MOV     #-1,R5          ; FLAG SUBUNIT AS NOT TESTED
     38 005457  100275                           2$:     MOV     R5,(R0)+        ; MOVE TO SUBUNIT POINTERS (NON-NEG IF TESTED)
     39 005460  117403                                   DEC     R3              ; DECREMENT COUNT
     40 005461  055451                                   BNE     1$              ; IF ALL SUBUNITS ARE FILLED IN, BRANCH
     41 005462                                          POP     R0              ; RESTORE R0
        005462  104267                                                                      MOV (SP)+,R0
     42 005463  107204  000004                           SUB     #4,R4           ; R4 NOW POINTS TO BEGINNING OF UNIT HANDLED
     43 005465  104302  004507                 NOU:      MOV     UMASK,R2        ; GET UNIT PORT MASK
     44 005467  110202                                   ROL     R2              ; ROTATE TO NEXT PORT
     45 005470  104020  004507                           MOV     R2,UMASK        ; SAVE MASK
     46 005472  105204  000004                           ADD     #4,R4           ; R4 POINTS TO NEXT UNIT INFORMATION
     47 005474  106204  005374                           CMP     #UNITS+16.,R4   ; SEE IF ALL UNITS SET UP
     48 005476  055402                                   BNE     ULOP            ; IF ALL UNITS NOT SETUP, BRANCH
     49 005477  104203  007702                           MOV     #FIRSTU,R3      ; COMPLETE RING
     50 005501  100703  177003                           MOV     R3,@LASTU       ; LAST UNIT NOW POINTS TO FIRST
     51 005503                                          ASSUME  U.NEXT,0
     52 005503  104207  007702                           MOV     #FIRSTU,R0      ; R0 POINTS TO FIRST UNIT PARAMETERS
     53 005505                                 SUSETL:   PUSH    R0              ; SAVE POINTER TO UNIT PARAMETERS
        005505  100467                                                                      MOV R0,-(SP)
     54                                         ;
```

```
                                        .SBTTL  BLDSUS - BUILD ALL SUBUNIT PARAMETER BLOCKS ON THIS UNIT
                                        ;BLDSUS
                                        ;
                                        ;       BUILD ALL SUBUNIT PARAMETERS
                                        ;
005506  104204  000001                        MOV     #U.SUBP,R4      ; R4 WILL POINT TO SUBUNIT POINTERS
005510  105074                                ADD     R0,R4           ; R4 POINTS TO SUBUNIT POINTERS
005511  104670  000063  004505                MOV     U.UNUM(R0),UCURSR       ; SAVE SUBUNIT NUMBER
005514  104205  000004                        MOV     #4,R5           ; MAXIMUM NUMBER OF SUBUNITS
005516  104147                        SLOP:   MOV     (R4),R0         ; R0 GETS SUBUNIT 'POINTER'
005517  076013                                BMI     NOSUN           ; IF SUBUNIT INACTIVE, BRANCH
005520  104070  004505                        MOV     R0,UCURSR       ; UCURSR GETS SUBUNIT NUMBER
005522                                        PUSH    <R3,R4,R5>      ; SAVE R3 - R5
   005522  100463                                                                       MOV R3,-(SP)
   005523  100464                                                                       MOV R4,-(SP)
   005524  100465                                                                       MOV R5,-(SP)
```

```
                                        .SBTTL  BLDBES - FIND HOW MANY WORDS NEEDED FOR THE BEGIN/END OR TRACK/GROUP SETS
                                        ;BLDBES
                                        ;
                                        ;       COMPUTE THE BEGIN/END SET AREA NEEDED
                                        ;
 005525  104300  004505  001104          MOV     UCURSR,HRQ.01   ; MOVE SUBUNIT NUMBER TO COMMUNICATION AREA
 005530  104207  060004                  MOV     #T4UPRM,R0      ; MOVE REQUEST NUMBER TO R0
 005532  021053                          CALL    HOSTRQ          ; REQUEST INFORMATION FROM HOST
 005533  102200  000040  001104          BIT     #BEUSED,HRQ.01  ; SEE IF BEGIN/END SETS ARE USED
 005536  055544                          BNE     2$              ; IF SO, BRANCH
 005537  104207  000020                  MOV     #S.TGSS+1,R0    ; MAXIMUM CONFIGURATION LENGTH
 005541  105307  001112                  ADD     HRQ.07,R0       ; ADD NUMBER OF T/G SETS
 005543  005555                          BR      BLDBB           ; BRANCH
 005544  104207  000013          2$:     MOV     #S.BESS,R0      ; MINIMUM CONFIGURATION WORD COUNT
 005546  104301  001106                  MOV     HRQ.03,R1       ; GET NUMBER OF BEGIN/END SETS
 005550  055552                          BNE     1$              ; IF NON-ZERO, BRANCH
 005551  115401                          INC     R1              ; MAKE R1 1
 005552  105011          1$:     ADD     R1,R1           ; MAKE B/E SETS * 2
 005553  105011                  ADD     R1,R1           ; MAKE B/E SETS * 4
 005554  105017                  ADD     R1,R0           ; ADD TO LENGTH OF SUBUNIT PARAMETERS
```

BLDBB - FIND HOW MANY WORDS NEEDED FOR THE BAD BLOCKS

```
      1                                          .SBTTL  BLDBB  - FIND HOW MANY WORDS NEEDED FOR THE BAD BLOCKS
      2 005555                                   BLDBB:
      3                                          ;
      4                                          ;       COUNT THE BAD BLOCK PARAMETERS
      5                                          ;
      6 005555                                           PUSH    R0              ; SAVE SUBUNIT LENGTH
        005555  100467                                                                   MOV R0,-(SP)
      7 005556  104300  004505  001104                   MOV     UCURSR,HRQ.01   ; MOVE SUBUNIT NUMBER TO COMMUNICATION AREA
      8 005561  104207  060005                           MOV     #T4BB1,R0       ; MOVE REQUEST NUMBER TO R0
      9 005563  021053                                   CALL    HOSTRQ          ; REQUEST INFORMATION FROM HOST
     10 005564                                           POP     R0              ; RESTORE SUBUNIT LENGTH
        005564  104267                                                                   MOV (SP)+,R0
     11 005565  104301  001104                           MOV     HRQ.01,R1       ; GET NUMBER OF BAD BLOCKS
     12 005567  104010  004510                           MOV     R1,NUMBB        ; SAVE NUMBER OF BAD BLOCKS
     13 005571  105011                                   ADD     R1,R1           ; MAKE BAD BLOCKS * 2
     14 005572  105017                                   ADD     R1,R0           ; ADD TO SUBUNIT DATA STRUCTURE LENGTH
     15 005573  024625                                   CALL    GETMEM          ; GET REQUESTED AMOUNT OF MEMORY
```

```
                                        .SBTTL  COPYSU - COPY ALL SUBUNIT PARAMETERS TO SUBUNIT BLOCK
                                        ;COPYSU
                                        ;
                                        ;       COPY SUBUNIT PARAMETRS TO SUBUNIT BLOCK
                                        ;
 005574                                         PUSH    R0              ; SAVE R0
 005574 100467                                                                  MOV R0,-(SP)
 005575 104300 004505 001104                    MOV     UCURSR,HRQ.01   ; MOVE SUBUNIT NUMBER TO COMMUNICATION AREA
 005600 104207 060004                           MOV     #T4UPRM,R0      ; MOVE REQUEST NUMBER TO R0
 005602 021053                                  CALL    HOSTRQ          ; REQUEST INFORMATION FROM HOST
 005603                                         POP     R0              ; RESTORE R0
 005603 104267                                                                  MOV (SP)+,R0
                                        ;
                                        ;       THESE ARE THE ONLY BITS THAT THE HOST SHOULD SEND TO THE TEST
                                        ;
         077177                         VALBIT  =       INITW!DCYLS!ECCCHK!RONLY!WONLY!RTRIES!       EUSED!TRACKS!WCHECK!WCHKAL!DAT
                                        ;
 005604 103200 100600 001104                    BIC     #^CVALBIT,HRQ.01        ; CLEAR ALL BUT VALID BITS (SUBUNIT PARAMETERS)
 005607 104301 001105                           MOV     HRQ.02,R1       ; GET PATTERN NUMBER
 005611 100671 000004                           MOV     R1,S.PAT(R0)    ; SAVE
 005613 104305 001104                           MOV     HRQ.01,R5       ; GET UNIT PARAMETERS
 005615 102205 020000                           BIT     #DCYLS,R5       ; SEE IF DIAGNOSTIC CYLINDERS ARE USED
 005617 015625                                  BEQ     4$              ; IF NOT, BRANCH
 005620 102205 004000                           BIT     #RONLY,R5       ; SEE IF READ ONLY
 005622 055625                                  BNE     4$              ; IF SO, BRANCH
 005623 101205 040000                           BIS     #INITW,R5       ; FLAG UNIS AS INITIALLY WRITTEN
 005625 102205 000040                   4$:     BIT     #BEUSED,R5      ; SEE IF BEGIN/END SETS USED
 005627 055672                                  BNE     3$              ; IF SO, BRANCH
 005630 100175                                  MOV     R5,(R0)         ; SAVE IN SUBUNIT AREA
 005631                                         ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
 005631 104201 000013                           MOV     #S.BESS,R1      ; R1 WILL POINT TO BEGIN/END SET AREA
 005633 105071                                  ADD     R0,R1           ; R1 POINTS TO TRACK/GROUP AREA
 005634 104202 001106                           MOV     #HRQ.03,R2      ; R2 POINTS TO START/END CYL
 005636 104225                                  MOV     (R2)+,R5        ; MOVE LIMITING CYLINDER TO SUBUNIT PARAMETERS
 005637 100615 000002                           MOV     R5,2(R1)        ; MOVE LIMITING CYLINDER TO SUBUNIT PARAMETERS
 005641 104225                                  MOV     (R2)+,R5        ; MOVE LIMITING CYLINDER TO SUBUNIT PARAMETERS
 005642 100615 000003                           MOV     R5,3(R1)        ; MOVE LIMITING CYLINDER TO SUBUNIT PARAMETERS
 005644 104225                                  MOV     (R2)+,R5        ; MOVE LIMITING CYLINDER TO SUBUNIT PARAMETERS
 005645 100115                                  MOV     R5,(R1)         ; MOVE LIMITING CYLINDER TO SUBUNIT PARAMETERS
 005646 104225                                  MOV     (R2)+,R5        ; MOVE LIMITING CYLINDER TO SUBUNIT PARAMETERS
 005647 100615 000001                           MOV     R5,1(R1)        ; MOVE LIMITING CYLINDER TO SUBUNIT PARAMETERS
 005651 105201 000004                           ADD     #4,R1           ; R1 POINTS AFTER LIMITING CYLINDERS
 005653 104224                                  MOV     (R2)+,R4        ; GET TRACK/GROUP COUNT
 005654 104225                          5$.     MOV     (R2)+,R5        ; GET WORD
 005655 100215                                  MOV     R5,(R1)+        ; SAVE
 005656 117404                                  DEC     R4              ; DECREMENT COUNT
 005657 055654                                  BNE     5$              ; IF INCOMPLETE, BRANCH
 005660 104415                                  MOV     -(R1),R5        ; GET LAST T/G
 005661 101205 100000                           BIS     #100000,R5      ; MARK AS END OF LIST
 005663 100115                                  MOV     R5,(R1)         ; SAVE
 005664 104201 000020                           MOV     #S.TGSS+1,R1    ; R1 WILL POINT TO START OF T/G LIST
 005666 105071                                  ADD     R0,R1           ; R1 POINTS TO START OF T/G LIST
 005667 105301 001112                           ADD     HRQ.07,R1       ; R1 NOW POINTS TO AFTER T/G'S (FOR BAD BLOCKS)
 005671 005730                                  BR      COPBB           ; BRANCH
 005672 104201 000013                   3$:     MOV     #S.BESS,R1      ; R1 WILL POINT TO BEGIN/END SETS
 005674 105071                                  ADD     R0,R1           ; R1 POINTS TO BEGIN/END SETS
 005675 104202 001106                           MOV     #HRQ.03,R2      ; POINT TO BEGIN/END SET COUNT
```

```
     56 005677  104224                          MOV     (R2)+,R4        ; GET COUNT, SEE IF NON-ZERO
     57 005700  055704                          BNE     2$              ; IF SO, BRANCH
     58 005701  115404                          INC     R4              ; SKIP COUNT
     59 005702  101205  000200                  BIS     #ONLYCL,R5      ; FLAG AS BEGIN/END CYLINDER SUPPLIED
     60 005704  100175                  2$:     MOV     R5,(R0)         ; SAVE IN SUBUNIT AREA
     61 005705                                  ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
     62 005705  104225                  1$:     MOV     (R2)+,R5        ; GET LO STARTING CYL
     63 005706  100615  000002                  MOV     R5,2(R1)        ; SAVE
     64 005710  104225                          MOV     (R2)+,R5        ; GET HI STARTING CYL
     65 005711  100615  000003                  MOV     R5,3(R1)        ; SAVE
     66 005713  104225                          MOV     (R2)+,R5        ; GET LO ENDING CYL
     67 005714  100115                          MOV     R5,(R1)         ; SAVE
     68 005715  104225                          MOV     (R2)+,R5        ; GET HI ORDER ENDING CYL
     69 005716  100615  000001                  MOV     R5,1(R1)        ; SAVE
     70 005720  105201  000004                  ADD     #4,R1           ; R1 POINTS AFTER BEGIN/END CYLINDERS
     71 005722  117404                          DEC     R4              ; DECREMENT COUNT
     72 005723  055705                          BNE     1$              ; LOOP AND COPY
     73 005724  104415                          MOV     -(R1),R5        ; MOVE EOL TO BEGIN/END LIST
     74 005725  101205  100000                  BIS     #100000,R5      ; SET HI BIT TO FLAG EOL
     75 005727  100215                          MOV     R5,(R1)+        ; SAVE
     76 005730                          COPBB:  PUSH    R0              ; SAVE R0
        005730  100467                                                                  MOV R0,-(SP)
     77 005731  104300  004505  001104          MOV     UCURSR,HRQ.01   ; MOVE TO COMMUNICATION AREA
     78 005734  104207  060005                  MOV     #T4BB1,R0       ; MOVE REQUEST NUMBER TO R0
     79 005736  021053                          CALL    HOSTRQ          ; REQUEST INFORMATION FROM HOST
     80 005737                                  POP     R0              ; RESTORE R0
        005737  104267                                                                  MOV (SP)+,R0
     81 005740  104202  001105                  MOV     #HRQ.02,R2      ; R2 POINTS AT BAD BLOCKS
     82 005742  104304  004510                  MOV     NUMBB,R4        ; R4 IS NUMBER OF BAD BLOCKS
     83 005744  016002                          BEQ     NOBB            ; IF NO BAD BLOCKS, BRANCH
     84 005745  100671  000012                  MOV     R1,S.BADP(R0)   ; MOVE POINTER TO BAD BLOCKS TO SUB PARAM
     85 005747  114004                          CLR     R4              ; START LOOP AT ZERO
     86 005750  106204  000016          BBLOP:  CMP     #14.,R4         ; SEE IF SECOND BLOCK NEEDED
     87 005752  055765                          BNE     NOEXTR          ; IF NOT, BRANCH
     88 005753                                  PUSH    R0              ; SAVE R0
        005753  100467                                                                  MOV R0,-(SP)
     89 005754  104300  004505  001104          MOV     UCURSR,HRQ.01   ; SAVE IN COMMUNICATION AREA
     90 005757  104207  060006                  MOV     #T4BB2,R0       ; LOAD REQUEST NUMBER
     91 005761  021053                          CALL    HOSTRQ          ; REQUEST FURTHER BLOCKS FROM HOST
     92 005762                                  POP     R0              ; RESTORE R0
        005762  104267                                                                  MOV (SP)+,R0
     93 005763  104202  001104                  MOV     #HRQ.01,R2      ; POINT R2 TO BAD BLOCKS
     94 005765  104225                  NOEXTR: MOV     (R2)+,R5        ; GET BAD BLOCK
     95 005766  100215                          MOV     R5,(R1)+        ; SAVE BAD BLOCK
     96 005767  104225                          MOV     (R2)+,R5        ; GET BAD BLOCK
     97 005770  100215                          MOV     R5,(R1)+        ; SAVE BAD BLOCK
     98 005771  115404                          INC     R4              ; INCREMENT R4
     99 005772  106304  004510                  CMP     NUMBB,R4        ; SEE IF ALL BLOCKS COPIED
    100 005774  055750                          BNE     BBLOP           ; IF NOT, BRANCH
    101 005775  104415                          MOV     -(R1),R5        ; GET LAST BAD BLOCK
    102 005776  101205  100000                  BIS     #100000,R5      ; FLAG AS EOL
    103 006000  100215                          MOV     R5,(R1)+        ; SAVE EOL
    104 006001  006004                          BR      CPYBEX          ; BRANCH
    105 006002  100674  000012          NOBB:   MOV     R4,S.BADP(R0)   ; FLAG AS NO BAD BLOCKS
    106 006004  114004                  CPYBEX: CLR     R4              ; CLEAR R4
    107 006005  100674  000007                  MOV     R4,S.SCHR(R0)   ; FLAG CHARACTERISTICS AS NOT YET DEFINED
    108                                 ;
```

```
    109 006007                                          POP     <R5,R4,R3>      ; RESTORE R3 - R5
        006007  104265                                                                      MOV (SP)+,R5
        006010  104264                                                                      MOV (SP)+,R4
        006011  104263                                                                      MOV (SP)+,R3
    110 006012  006015                                  BR      BSUSEX          ; BRANCH
    111 006013  104207  177777          NOSUN:         MOV     #-1,R0          ; TO FLAG SUBUNIT AS INACTIVE
    112 006015  100247                   BSUSEX:        MOV     R0,(R4)+        ; MOVE POINTER TO SUB PARAM TO UNIT PARAMS
    113 006016  115400  004505                          INC     UCURSR          ; MOVE TO NEXT SUBUNIT
    114 006020  117405                                  DEC     R5              ; DECREMENT COUNT
    115 006021  055516                                  BNE     SLOP            ; IF UNEXPIRED, BRANCH
    116 006022                                          POP     R0              ; RESTORE UNIT POINTER
        006022  104267                                                                      MOV (SP)+,R0
    117 006023  104177                                  MOV     (R0),R0         ; GO TO NEXT UNIT
    118 006024                                          ASSUME  U.NEXT,0
    119 006024  103207  170000                          BIC     #^CHBHINB,R0    ; CLEAR UNUSED BITS
    120 006026  106207  007702                          CMP     #FIRSTU,R0      ; SEE IF THE ENTIRE RING IS SET UP
    121 006030  055505                                  BNE     SUSETL          ; IF NOT, BRANCH
    122 006031  000000                                  RETURN
    123                                                 .IF     LE,MAXADR-.
    124                                  MAXADR         =       .+1
    125                                                 .ENDC
```

```
                                        .SBTTL  ********** NON-LOADED OVERLAY, ERROR MESSAGES
                                        ;*****************************************************************************
                                        ;*****************************************************************************
                                        ;*****************************************************************************
                                        ;*****************************************************************************
                                        ;
                                        ;       ERROR MESSAGE FORMAT BUFFERS (NEVER LOADED INTO UDA)
                                        ;
006032                                  DMOVLY  MS,0                    ; ERROR MESSAGE OVERLAY
006032  000105                         .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                        .NLIST  BEX
000000      042     101     124 ER1:    .ASCII  \'ATTN ASSERTED DURING SEEK'N\
000016      042     123     105         .ASCII  \'SEEK FROM GRP 'D8'' CYL 'D28'' TO GRP 'D8'' CYL 'D28NR1\
000050      000                         .BYTE   0
000051      042     122     105 SER22:  .ASCII  \'REAL TIME STATE 'H16N\
000064      042     123     124         .ASCII  \'STATUS (R TO L): 'H16S2H16S2H16S2H16S2H16S2H16S2H16N\
000116      000                         .BYTE   0
000117      042     101     124 ER2:    .ASCII  \'ATTN ASSERTED UNEXPECTEDLY, ASYNC DRIVE ERROR OR LOGGABLE'N\
000155      042     111     116         .ASCII  \'INFORMATION'NR1\
000165      000                         .BYTE   0
000166      042     123     105 ER3:    .ASCII  \'SEEK DID NOT COMPLETE, NEITHER ATTN OR R/W RDY WAS ASSERTED'N\
000225      042     102     105         .ASCII  \'BEFORE TIMEOUT'N\
000235      042     123     105         .ASCII  \'SEEK FROM GRP 'D8'' CYL 'D28'' TO GRP 'D8'' CYL 'D28NR1\
000270      000                         .BYTE   0
000271      042     122     103 ER4:    .ASCII  \'RCT AREA CORRUPTED, COULD NOT FIND REPLACEMENT FOR'N\
000323      042     114     102         .ASCII  \'LBN 'R1\
000327      042     101     124         .ASCII  \'ATTEMPTED TO READ RCT LBN 'D28N\
000347      042     123     105         .ASCII  \'SEARCING FOR LBN'S10D28N\
000364      000                         .BYTE   0
000365      042     110     105 ER5:    .ASCII  \'HEADER NOT FOUND DURING WRITE'N\
000405      122     061     042         .ASCII  \R1'BN 'D28NR3\
000413      042     123     105         .ASCII  \'SECTORS FROM INDEX 'D8'' TRK 'D8'' GRP 'D8'' CYL 'D28N\
000445      042     117     122         .ASCII  \'ORIGIN OF SEEK: GRP 'D8'' CYL 'D28N\
000467      000                         .BYTE   0
000470      042     101     124 SER19:  .ASCII  \'ATTEMPT 'D3N\
000476      122     061     042         .ASCII  \R1'BN 'D28NR3\
000505      042     123     105         .ASCII  \'SECTORS FROM INDEX 'D8'' TRK 'D8'' GRP 'D8'' CYL 'D28N\
000537      042     117     122         .ASCII  \'ORIGIN OF SEEK: GRP 'D8'' CYL 'D28NR1\
000561      000                         .BYTE   0
000562      042     123     105 ER6:    .ASCII  \'SELECT TRACK AND WRITE LEVEL 1 CMD NOT EXECUTED'NR1\
000614      000                         .BYTE   0
000615      042     105     103 ER7:    .ASCII  \'ECC DETECTED ERROR'NR1\
000630      000                         .BYTE   0
000631      042     105     103 ER8:    .ASCII  \'ECC DETECTED ERROR, BUT CORRECTION FAILED'NR1\
000660      000                         .BYTE   0
000661      042     122     105 SER21:  .ASCII  \'RETRY 'D4N\
000666      042     105     122         .ASCII  \'ERROR RECOVERY LEVEL 'D8N\
000703      122     061     042         .ASCII  \R1'BN 'D28NR3\
000712      042     123     105         .ASCII  \'SECTORS FROM INDEX 'D8'' TRK 'D8'' GRP 'D8'' CYL 'D28N\
000744      000                         .BYTE   0
000745      042     105     103 ER9:    .ASCII  \'ECC CORRECTIONS EXCEED THRESHOLD'NR1\
000767      000                         .BYTE   0
000770      042     105     103 ER10:   .ASCII  \'ECC CORRECTION SUCCEEDED, BUT EDC DETECTS ERROR'NR1\
001022      042     105     104         .ASCII  \'EDC COMPUTED 'D16N\
001033      042     105     104         .ASCII  \'EDC READ'S5016N\
001043      000                         .BYTE   0
001044      042     105     122 ER11:   .ASCII  \'ERROR RECOVERY TRIED ALL LEVELS WITHOUT SUCCESS'N\
```

```
     57 001075      122      061      042            .ASCII  \R1'BN 'D28NR3\
     58 001103      042      107      122            .ASCII  \"GRP 'D8" CYL 'D28N\
     59 001115      000                              .BYTE   0
     60 001116      042      104      101  ER12:     .ASCII  \"DATA COMPARISON FAILED'NR1\
     61 001133      122      061      042            .ASCII  \R1'BN 'D28NR3\
     62 001142      042      123      105            .ASCII  \"SECTORS FROM INDEX 'D8" TRK 'D8" GRP 'D8" CYL 'D28N\
     63 001174      042      120      101            .ASCII  \"PATTERN NUMBER  'D4N\
     64 001206      042      117      106            .ASCII  \"OFFSET OF ERROR WITHIN BUFFER: 'D8N\
     65 001230      042      117      106            .ASCII  \"OFFSET OF ERROR WITHIN DISPLAYED LIST: 'D8" (1ST WORD OFFSET 0)'N\
     66 001271      123      064      117            .ASCII  \S4016S4016S4016S4016S4016S4016N\
     67 001311      123      064      117            .ASCII  \S4016S4016S4016S4016S4016S4016N\
     68 001330      000                              .BYTE   0
     69 001331      042      105      103  SER24:    .ASCII  \"ECC OR EDC HAD DETECTED ERROR IN BUFFER'N\
     70 001356      000                              .BYTE   0
     71 001357      042      105      103  SER25:    .ASCII  \"ECC OR EDC HAD <<NOT>> DETECTED ERROR IN BUFFER'N\
     72 001410      000                              .BYTE   0
     73 001411      042      104      122  ER13:     .ASCII  \"DRIVE NOT ONLINE TO UDA, AND NOT SPINABLE'N\
     74 001437      000                              .BYTE   0
     75 001440      042      125      116  ER14:     .ASCII  \"UNABLE TO COMPLETE SEEK -- TRIED 3 TIMES'N\
     76 001465      122      061      042            .ASCII  \R1'BN 'D28NR3\
     77 001474      042      107      122            .ASCII  \"GRP 'D8" CYL 'D28N\
     78 001505      000                              .BYTE   0
     79 001506      042      123      105  ER15:     .ASCII  \"SEEK REQUIRED 'D2" RETRIES BEFORE COMPLETING'N\
     80 001535      122      061      042            .ASCII  \R1'BN 'D28NR3\
     81 001544      042      107      122            .ASCII  \"GRP 'D8" CYL 'D28N\
     82 001555      000                              .BYTE   0
     83 001556      042      105      122  ER16:     .ASCII  \"ERRORS DURING DRIVE INITIALIZATION AND SETUP'NR1\
     84 001606      000                              .BYTE   0
     85 001607      042      116      117  ER17:     .ASCII  \"NO VALID STATE FROM DRIVE'N\
     86 001625      042      116      117            .ASCII  \"NO DRIVE CLOCKS'N\
     87 001636      000                              .BYTE   0
     88 001637      042      116      117  ER17A:    .ASCII  \"NO VALID STATE FROM DRIVE'N\
     89 001655      042      110      101            .ASCII  \"HARD PARITY OR PULSE ERROR FOR 1/2 A SECOND'N\
     90 001704      000                              .BYTE   0
     91 001705      042      101      124  ER18:     .ASCII  \"ATTEMPT TO WRITE ON WRITE PROTECTED DRIVE'N\
     92 001733      042      105      122            .ASCII  \"ERROR CODE RETURNED FROM UDA: 'D16NR1\
     93 001756      000                              .BYTE   0
     94 001757      042      110      105  ER19:     .ASCII  \"HEADER NOT FOUND DURING READ'N\
     95 001776      122      061      042            .ASCII  \R1'BN 'D28NR3\
     96 002005      042      123      105            .ASCII  \"SECTORS FROM INDEX 'D8" TRK 'D8" GRP 'D8" CYL 'D28N\
     97 002037      042      117      122            .ASCII  \"ORIGIN OF SEEK: GRP 'D8" CYL 'D28N\
     98 002060      000                              .BYTE   0
     99 002061      042      123      105  ER20:     .ASCII  \"SELECT TRACK AND READ LEVEL 1 CMD NOT SENT'NR1\
    100 002110      000                              .BYTE   0
    101 002111      042      104      122  ER21:     .ASCII  \"DRIVE NOT FORMATTED IN 512 BYTE MODE -- UNABLE TO TEST'N\
    102 002145      042      130      102            .ASCII  \"XBN 0 MODE WORD: 'D16N\
    103 002161      000                              .BYTE   0
    104 002162      042      125      116  ER23:     .ASCII  \"UNABLE TO CONTINUE TESTING'NR1R1\
    105 002202      000                              .BYTE   0
    106 002203      042      122      125  SER42:    .ASCII  \"RUN/STOP SWITCH OUT'N\
    107 002216      000                              .BYTE   0
    108 002217      042      123      120  SER43:    .ASCII  \"SPINDLE DROPPED READY'N\
    109 002233      000                              .BYTE   0
    110 002234      042      120      117  SER44:    .ASCII  \"PORT SWITCH OUT'N\
    111 002245      000                              .BYTE   0
    112 002246      042      120      101  SER45:    .ASCII  \"PARITY ERRORS FOR MORE THAN A HALF SECOND'N\
    113 002274      000                              .BYTE   0
```

```
    114 002275      042     105     104  ER24:   .ASCII  \'EDC DETECTED ERROR BUT ECC DID NOT'NR1\
    115 002320      042     105     104          .ASCII  \'EDC COMPUTED '0I6'' EDC READ '0I6N\
    116 002341      000                          .BYTE   0
    117 002342      042     127     122  ER25:   .ASCII  \'WRITE ATTEMPTED MAXIMUM TIMES'N\
    118 002362      122     061     042          .ASCII  \R1'BN '0D28NR3\
    119 002370      000                          .BYTE   0
    120 002371      042     122     105  ER26:   .ASCII  \'READ ATTEMPTED MAXIMUM TIMES'N\
    121 002410      122     061     042          .ASCII  \R1'BN '0D28NR3\
    122 002417      000                          .BYTE   0
    123 002420      042     102     117  ER28:   .ASCII  \'BOTH READ ONLY <AND> WRITE ONLY BITS SET -- HOST ERROR'N\
    124 002454      000                          .BYTE   0
    125 002455      042     125     116  ER33:   .ASCII  \'UNABLE TO CORRECTLY READ OVERLAY '0I6NR1\
    126 002501      000                          .BYTE   0
    127 002502      042     124     110  SER39:  .ASCII  \'THIS UDA AND ALL DRIVES ATTACHED WILL BE REMOVED FROM TESTING'N\
    128 002542      000                          .BYTE   0
    129 002543      042     123     105  ER34:   .ASCII  \'SERDES OVERRUN ERROR DURING READ'NR1\
    130 002565      000                          .BYTE   0
    131 002566      042     104     101  ER35:   .ASCII  \'DATA OR STATE CLOCK TIMEOUT DURING READ'NR1\
    132 002614      000                          .BYTE   0
    133 002615      042     104     101  ER36:   .ASCII  \'DATA SYNC TIMEOUT DURING READ'NR1\
    134 002636      000                          .BYTE   0
    135 002637      042     122     057  ER37:   .ASCII  \'R/W RDY DROPPED BEFORE/DURING READ'NR1\
    136 002662      000                          .BYTE   0
    137 002663      042     122     103  ER38:   .ASCII  \'RCVR RDY DROPPED BEFORE/DURING READ'NR1\
    138 002707      000                          .BYTE   0
    139 002710      042     101     114  ER40:   .ASCII  \'ALL COPIES OF RCT READ WITH ERROR, SEARCHING FOR'N\
    140 002741      042     114     102          .ASCII  \'LBN '0R1\
    141 002745      042     114     101          .ASCII  \'LAST RCT LBN SEARCHED '0D28N\
    142 002763      042     123     105          .ASCII  \'SEARCING FOR LBN'S6D28N\
    143 002777      000                          .BYTE   0
    144 003000      042     103     117  ER41:   .ASCII  \'COULD NOT FIND REPLACEMENT FOR'N\
    145 003020      042     114     102          .ASCII  \'LBN '0R1\
    146 003024      042     114     102          .ASCII  \'LBN TO REPLACE '0D28N\
    147 003037      000                          .BYTE   0
    148 003040      042     124     110  SER26:  .ASCII  \'THAT WAS REVECTORED'N\
    149 003053      000                          .BYTE   0
    150 003054      042     127     111  SER32:  .ASCII  \'WITH HEADER NOT FOUND'N\
    151 003070      000                          .BYTE   0
    152 003071      042     124     111  ER42:   .ASCII  \'TIMEOUT WAITING FOR SECTOR OR INDEX PULSE'N\
    153 003117      042     107     122          .ASCII  \'GRP '0D8'' CYL '0D28NR1\
    154 003131      000                          .BYTE   0
    155 003132      042     123     105  ER44:   .ASCII  \'SEEK OR HEAD SELECT ERROR DETECTED DURING WRITE'NR1\
    156 003164      000                          .BYTE   0
    157 003165      042     123     105  ER45:   .ASCII  \'SEEK OR HEAD SELECT ERROR DETECTED DURING READ'NR1\
    158 003216      000                          .BYTE   0
    159 003217      042     104     101  ER47:   .ASCII  \'DATA OR STATE CLOCK TIMEOUT DURING WRITE'NR1\
    160 003245      000                          .BYTE   0
    161 003246      042     122     057  ER48:   .ASCII  \'R/W RDY DROPPED BEFORE/DURING WRITE'NR1\
    162 003272      000                          .BYTE   0
    163 003273      042     122     103  ER49:   .ASCII  \'RCVR RDY DROPPED BEFORE/DURING WRITE'NR1\
    164 003317      000                          .BYTE   0
    165 003320      122     061     042  ER50:   .ASCII  \R1'BEGIN/END SET STARTING BLOCK NUMBER GREATER THAN ENDING BLOCK NUMBER'N\
    166 003364      000                          .BYTE   0
    167 003365      122     061     042  ER51:   .ASCII  \R1''THE BEGIN/END SETS GIVEN OVERLAP'N\
    168 003407      000                          .BYTE   0
    169 003410      122     061     042  ER52:   .ASCII  \R1'BEGIN/END SET ENDING BLOCK NUMBER EXCEEDS MAXIMUM'N\
    170 003443      042     115     101          .ASCII  \'MAXIMUM BLOCK NUMBER ON DEVICE IS '0D28N\
```

```
    171 003467  000                                 .BYTE   0
    172 003470  122  061  042  ER53:                .ASCII  \R1'DUPLICATE BAD BLOCKS'N\
    173 003504  000                                 .BYTE   0
    174 003505  122  061  042  ER54:                .ASCII  \R1'BAD BLOCK NUMBER EXCEEDS MAXIMUM. MAXIMUM BLOCK NUMBER'N\
    175 003542  042  117  116                       .ASCII  \''ON DEVICE IS 'D28N\
    176 003554  000                                 .BYTE   0
    177 003555  122  061  042  ER55:                .ASCII  \R1''STARTING CYLINDER GREATER THAN ENDING CYLINDER'N\
    178 003606  000                                 .BYTE   0
    179 003607  122  061  042  ER56:                .ASCII  \R1'RANDOM AND         AL SEEKS CAN NOT BE MIXED WITHIN A UNIT'N\
    180 003646  000                                 .BYTE   0
    181 003647  122  061  042  ER57:                .ASCII  \R1'OVERFLOW WHEN CALCULATING THE L/DBN FROM THE GIVEN CYLINDER'N\
    182 003707  042  103  131                       .ASCII  \''CYLINDER TOO LARGE'N\
    183 003721  000                                 .BYTE   0
    184 003722  122  061  122  ER58:                .ASCII  \R1R1'' EXCEEDS MAXIMUM FOR DEVICE. MAXIMUM IS 'D8N\
    185 003752  000                                 .BYTE   0
    186 003753  122  061  042  ER59:                .ASCII  \R1''TWO IDENTICAL 'R1''S'N\
    187 003767  000                                 .BYTE   0
    188 003770  122  061  122  ER62:                .ASCII  \R1R1'BN COMPUTED FROM END CYLINDER GIVEN EXCEEDS MAXIMUM 'R1'BN ON'N\
    189 004032  042  104  105                       .ASCII  \'DEVICE - CYLINDER TOO LARGE'N\
    190 004051  000                                 .BYTE   0
    191 004052  042  117  120  SER20:               .ASCII  \''OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT'N\
    192 004117  000                                 .BYTE   0
    193 004120  042  122  105  ER63:                .ASCII  \'REAL TIME STATE RECEIVE ERROR DURING WRITE'NR1\
    194 004147  000                                 .BYTE   0
    195 004150  042  122  105  ER64:                .ASCII  \'REAL TIME STATE RECEIVE ERROR DURING READ'NR1\
    196 004177  000                                 .BYTE   0
    197 004200  042  125  116  ER68:                .ASCII  \'UNKNOWN ERROR CODE DURING WRITE'NR1\
    198 004222  000                                 .BYTE   0
    199 004223  042  125  116  ER69:                .ASCII  \'UNKNOWN ERROR CODE DURING READ'NR1\
    200 004244  000                                 .BYTE   0
    201 004245  042  105  122  SER36:               .ASCII  \'ERROR CODE RETURNED 'D16NR1\
    202 004263  000                                 .BYTE   0
    203 004264  042  124  111  ER70:                .ASCII  \''TIMEOUT OF SEND'NR1R1\
    204 004277  000                                 .BYTE   0
    205 004300  042  124  111  ER71:                .ASCII  \''TIMEOUT OF RECEIVE'NR1R1\
    206 004314  000                                 .BYTE   0
    207 004315  042  106  111  ER72:                .ASCII  \''FIRST WORD RECEIVED WAS NOT START FRAME'NR1R1\
    208 004344  000                                 .BYTE   0
    209 004345  042  106  122  ER73:                .ASCII  \''FRAMING ERROR ON LEVEL 0 RECEIVE'NR1R1\
    210 004370  000                                 .BYTE   0
    211 004371  042  103  110  ER74:                .ASCII  \''CHECKSUM ERROR ON LEVEL 0 RECEIVE'NR1R1\
    212 004415  000                                 .BYTE   0
    213 004416  042  102  125  ER75:                .ASCII  \''BUFFER SIZE SMALLER THAN LEVEL 2 RESPONSE'NR1R1\
    214 004446  000                                 .BYTE   0
    215 004447  042  122  105  ER76:                .ASCII  \'RESPONSE OF LEVEL 2 CMD NOT AS EXPECTED'NR1\
    216 004475  042  105  130                       .ASCII  \'EXPECTED RESPONSE 'H8N\
    217 004510  042  122  105                       .ASCII  \'RESPONSE RECEIVED 'H8NR1\
    218 004525  000                                 .BYTE   0
    219 004526  042  104  122  ER77:                .ASCII  \''DRIVE NEVER DEASSERTED RECEIVER READY AFTER SEND'NR1R1\
    220 004561  000                                 .BYTE   0
    221 004562  042  125  116  ER78:                .ASCII  \''UNKNOWN ERROR CODE RETURNED FROM LEVEL 2 RECEIVE'NR1\
    222 004614  042  105  122                       .ASCII  \'ERROR CODE RETURNED 'D16NR1\
    223 004632  000                                 .BYTE   0
    224 004633  042  101  124  SER0:                .ASCII  \''ATTEMPTING TO GET STATUS'N\
    225 004650  000                                 .BYTE   0
    226 004651  042  101  124  SER1:                .ASCII  \''ATTEMPTING DRIVE CLEAR CMD'N\
    227 004667  000                                 .BYTE   0
```

```
    228 004670      042     101     124  SER2:    .ASCII  \"ATTEMPTING TO BRING DRIVE ONLINE'N\
    229 004711      000                           .BYTE   0
    230 004712      042     101     124  SER3:    .ASCII  \"ATTEMPTING TO CHANGE MODE'N\
    231 004730      000                           .BYTE   0
    232 004731      042     101     124  SER4:    .ASCII  \"ATTEMPTING ERROR RECOVERY CMD'N\
    233 004751      000                           .BYTE   0
    234 004752      042     101     124  SER5:    .ASCII  \"ATTEMPTING TO GET SUBUNIT CHAR'N\
    235 004772      000                           .BYTE   0
    236 004773      042     101     124  SER6:    .ASCII  \"ATTEMPTING TO SPIN UP DRIVE'N\
    237 005012      000                           .BYTE   0
    238 005013      042     101     124  SER7:    .ASCII  \"ATTEMPTING TO RECALIBRATE'N\
    239 005031      000                           .BYTE   0
    240 005032      042     101     124  SER8:    .ASCII  \"ATTEMPTING TO GET COMMON CHAR'N\
    241 005052      000                           .BYTE   0
    242 005053      042     101     124  SER9:    .ASCII  \"ATTEMPTING TO ISSUE SEEK'N\
    243 005070      000                           .BYTE   0
    244 005071      042     125     116  ER1000:  .ASCII  \"UNABLE TO FIND REQUESTED DRIVE FOR TESTING'N\
    245 005117      042     124     110          .ASCII  \"THE FOLLOWING IS VISIBLE ON THE PORTS'N\
    246 005143      042     125     104          .ASCII  \"UDA PORT 0 -- 'R1\
    247 005154      042     125     104          .ASCII  \"UDA PORT 1 -- 'R1\
    248 005165      042     125     104          .ASCII  \"UDA PORT 2 -- 'R1\
    249 005176      042     125     104          .ASCII  \"UDA PORT 3 -- 'R1\
    250 005207      000                           .BYTE   0
    251 005210      042     116     117  SER10:   .ASCII  \"NO DRIVE ATTACHED'N\
    252 005222      000                           .BYTE   0
    253 005223      042     122     103  SER11:   .ASCII  \"RCVR RDY NEVER ASSERTED'N\
    254 005240      000                           .BYTE   0
    255 005241      042     124     111  SER12:   .ASCII  \"TIMEOUT OF SEND'N\
    256 005252      000                           .BYTE   0
    257 005253      042     124     111  SER13:   .ASCII  \"TIMEOUT OF RECEIVE'N\
    258 005265      000                           .BYTE   0
    259 005266      042     106     111  SER14:   .ASCII  \"FIRST WORD RECEIVED WAS NOT START FRAME'N\
    260 005313      000                           .BYTE   0
    261 005314      042     106     122  SER15:   .ASCII  \"FRAMING ERROR ON LEVEL 0 RECEIVE'N\
    262 005335      000                           .BYTE   0
    263 005336      042     103     110  SER16:   .ASCII  \"CHECKSUM ERROR ON LEVEL 0 RECEIVE'N\
    264 005360      000                           .BYTE   0
    265 005361      042     122     105  SER17:   .ASCII  \"RESPONSE LONGER THAN EXPECTED FOR GET STATUS CMD'N\
    266 005412      000                           .BYTE   0
    267 005413      042     104     122  SER18:   .ASCII  \"DRIVE 'R1\
    268 005420      000                           .BYTE   0
    269 005421      104     061     062  SER18D:  .ASCII  \D12'', ''\       ; NOTE: THE STRING WITHIN THE \\'S <<MUST>>
    270 005425      104     061     062  SER18C:  .ASCII  \D12'', ''\       ; BE AN EVEN NUMBER OF CHAR, BECAUSE THE
    271 005431      104     061     062  SER18B:  .ASCII  \D12'', ''\       ; LABELS WILL FORCE WORD ALINGMENT, LEAVING
    272 005435      104     061     062  SER18A:  .ASCII  \D12N\            ; JUNK IN THE LAST BYTE. (SER18A OK TO BE ODD)
    273 005437      000                           .BYTE   0
    274 005440      042     104     122  SER40:   .ASCII  \"DRIVE NOT AVAILABLE TO THIS UDA'N\
    275 005461      000                           .BYTE   0
    276 005462      042     125     116  SER41:   .ASCII  \"UNSPINABLE DRIVE 'R1\
    277 005474      000                           .BYTE   0
    278 005475      042     122     105  RBNTXT:  .ASCII  \"REVECTORED TO RBN 'D28N\
    279 005511      000                           .BYTE   0
    280 005512      042     124     122  TRK$:    .ASCII  \"TRACK'\
    281 005515      000                           .BYTE   0
    282 005516      042     107     122  GRP$:    .ASCII  \"GROUP'\
    283 005521      000                           .BYTE   0
    284 005522      042     114     042  L$:      .ASCII  \'L'\
```

```
    285 005523     000                                .BYTE   0
    286 005524     042     104     042  D$:           .ASCII  \'D'\
    287 005525     000                                .BYTE   0
    288 005526     042     122     103  RCTL$:        .ASCII  \'RCT L'\
    289 005531     000                                .BYTE   0
```

```
                                        .SBTTL  INFORMATIONAL MESSGES
                                        ;
                                        ;       MESSAGES
                                        ;
005532  116     042     111     MS1:    .ASCII  \N''INITIAL WRITE COMPLETE'N\
005547  000                             .BYTE   0
005550  116     042     124     MS2:    .ASCII  \N''THE PREVIOUS DEVICE FATAL WILL CAUSE THE FOLLOWING DRIVES'N\
005606  042     124     117             .ASCII  \''TO BE DROPPED: 'R1\
005620  000                             .BYTE   0
005621  116     042     101     MS3:    .ASCII  \N''A CORRECTABLE ECC ERROR EXISTS IN 'R1'BN 'D28N\
005651  042     123     105             .ASCII  \''SECTORS FROM INDEX 'D8'' TRK 'D8'' GRP 'D8'' CYL 'D28N\
005703  000                             .BYTE   0
005704  116     042     122     MS4:    .ASCII  \N'READ ONLY DRIVE, INITIAL WRITE WILL NOT BE PERFORMED'N\
005740  000                             .BYTE   0
```

```
                                .SBTTL  ********** OVERLAY MODULE SETUP - OPERATION TO BE DONE THIS PASS
                                        DMOVLY  SU,AREA0
005740
005741  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
                                ;
000001                          SETUP   =       1               ; SETUP OVERLAY
                                ;
                                ;       SETUP CLEARS ALL ERROR BITS, THE ERROR COUNT, FINDS THE NEXT
                                ;       LBN TO BE READ OR WRITTEN, DESIDES TO READ OR WRITE, CALCULATES
                                ;       THE CYLINDER, TRACK AND GROUP THE LBN RESIDES ON; THEN CALLS
                                ;       SEEK.
                                ;
                                        .ENABL  LSB
004422  114002                          CLR     R2              ; NO ERRORS
004423  104657  000024                  MOV     U.CSEC(R5),R0   ; GET TOTAL NUMBER OF SECTORS ALLREADY R/W
004425  105657  000021                  ADD     U.NSEC(R5),R0   ; ADD HOW MANY R/W THIS PASS
004427  106657  000023                  CMP     U.TSEC(R5),R0   ; SEE IF ALL DONE
004431  014451                          BEQ     2$              ; IF SO, BRANCH (NEW OPERATION)
004432  100657  000024                  MOV     R0,U.CSEC(R5)   ; SAVE CURRENT COUNT
004434  104657  000053                  MOV     U.CBN(R5),R0    ; GET LBN
004436  105657  000021                  ADD     U.NSEC(R5),R0   ; ADD NUMBER OF SECTORS R/W THIS PASS
004440  100657  000053                  MOV     R0,U.CBN(R5)    ; SAVE
004442  044523                          BCC     9$              ; IF NO CARRY, BRANCH
004443  104657  000054                  MOV     U.CBN+1(R5),R0  ; GET HI LBN
004445  115407                          INC     R0              ; PROPOGATE CARRY
004446  100657  000054                  MOV     R0,U.CBN+1(R5)  ; SAVE
004450  004523                          BR      9$              ; BRANCH
004451  104657  000046          2$:     MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
004453  102207  000010                  BIT     #WCHECK,R0      ; SEE IF WRITE CHECK IS TO BE DONE
004455  054504                          BNE     5$              ; IF SO, BRANCH
004456  104141                          MOV     (R4),R1         ; GET SUBUNIT PARAMETERS
004457                                  ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
004457  102201  000100                  BIT     #       1       ; SEE IF          AL SEEKS
004461  054471                          BNE     6$              ; IF SO, BRANCH
004462  102207  040000                  BIT     #INITW,R0       ; SEE IF INITIAL WRITE IN PROGRESS
004464  054471                          BNE     6$              ; IF SO, BRANCH
004465  101207  004000                  BIS     #NEWSUB,R0      ; FLAG AS TO GO ONTO A NEW SUBUNIT
004467  100657  000046                  MOV     R0,U.PARM(R5)   ; SAVE
004471  104207  000002          6$:     MOV     #NEWOP,R0       ; SET UP A NEW OPERATION
004473  114000  002217                  CLR     SCR1            ; CLEAR SCRATCH AREA1 (NO NEW SUBUNIT FLAG)
004475  104651  000023                  MOV     U.TSEC(R5),R1   ; GET TOTAL NUMBER OF SECTORS TO BE WRIITEN LAST OP
004477  054606                          BNE     4$              ; IF NOT FIRST TIME, BRANCH
004500  104200  177777  002217          MOV     #-1,SCR1        ; FLAG AS NEW SUBUNIT (WITHOUT GOING ONTO NEW SUBUNIT)
004503  004606                          BR      4$              ; EXIT
004504  103207  000110          5$:     BIC     #WCHECK!REDWRT,R0       ; CLEAR WRITE CHECK, MAKE OPERATION READ
004506  100657  000046                  MOV     R0,U.PARM(R5)   ; SAVE
004510  104657  000051                  MOV     U.MBN(R5),R0    ; GET LO LBN
004512  100657  000053                  MOV     R0,U.CBN(R5)    ; SAVE
004514  104657  000052                  MOV     U.MBN+1(R5),R0  ; GET HI LBN
004516  100657  000054                  MOV     R0,U.CBN+1(R5)  ; SAVE
004520  114002                          CLR     R2              ; NO ERRORS
004521  100652  000024                  MOV     R2,U.CSEC(R5)   ; SECTORS R/W IS ZERO
```

```
     73 004523  104657  000023          9$:     MOV     U.TSEC(R5),R0   ; GET TOTAL NUMBER OF SECTORS TO R/W
     74 004525  107657  000024                  SUB     U.CSEC(R5),R0   ; SUBTRACT HOW MANY DONE
     75 004527  104651  000046                  MOV     U.PARM(R5),R1   ; GET UNIT PARAMETERS
     76 004531  102201  000200                  BIT     #RBNBN,R1       ; SEE IF AN RBN IS BEING HANDLED
     77 004533  014537                          BEQ     1$              ; IF NOT, BRANCH
     78 004534  104207  000001                  MOV     #1,R0           ; ONLY READ/WRITE ONE SECTOR
     79 004536  004555                          BR      8$              ; BRANCH
     80 004537  102201  000100          1$:     BIT     #REDWRT,R1      ; SEE IF WRITE IS IN PROGRESS
     81 004541  054550                          BNE     7$              ; IF SO, BRANCH
     82 004542  106307  002233                  CMP     SECMAX,R0       ; SEE IF TRYING TO READ MORE THAN POSSIBLE
     83 004544  044555                          BCC     8$              ; IF NOT, BRANCH
     84 004545  104307  002233                  MOV     SECMAX,R0       ; SET R0 TO MAXIMUM POSSIBLE SECTORS TO READ
     85 004547  004555                          BR      8$              ; BRANCH
     86 004550  106307  002234          7$:     CMP     CHNMAX,R0       ; SEE IF TRYING TO WRITE MORE THAN POSSIBLE
     87 004552  044555                          BCC     8$              ; IF NOT, BRANCH
     88 004553  104307  002234                  MOV     CHNMAX,R0       ; SET R0 TO MAXIMUM POSSIBLE SECTORS TO WRITE
     89 004555  100657  000022          8$:     MOV     R0,U.MSEC(R5)   ; SAVE IN NUMBER OF SECTORS TO R/W THIS PASS
     90 004557  104653  000031          3$:     MOV     U.MLEV(R5),R3   ; GET MAXIMUM ERROR REVCOVERY LEVEL
     91 004561  100653  000027                  MOV     R3,U.ELEV(R5)   ; STORE IN CURRENT LEVEL
     92 004563  104203  177777                  MOV     #-1,R3          ; START RETRIES AT 0
     93 004565  100653  000012                  MOV     R3,U.RWTO(R5)   ; SAVE IN UNIT PARAMETER AREA
     94                             ;           NOTE:   R2 IS ZERO AT THIS POINT
     95 004567  100652  000021                  MOV     R2,U.NSEC(R5)   ; NO SECTORES SUCCESSFULLY READ/WRITTEN
     96 004571  100652  000010                  MOV     R2,U.RRTY(R5)   ; ZERO READ RETRY VALUE
     97 004573  104207  000023                  MOV     #SEEK,R0        ; POINT TO SEEK AS NEXT OPERATION
     98 004575  024203                          CALL    ENABLE          ; ENABLE ERROR RECOVERY
     99 004576  102203  030000                  BIT     #LEVUSD!NXTLEV,R3       ; SEE IF ERROR RECOVERY LEVELS USED
    100 004600  014606                          BEQ     4$              ; IF NOT, BRANCH
    101 004601  024227                          CALL    GOSEEK          ; CAUSE A SEEK TO RESET ERROR RECOVERY
    102 004602  103203  030000                  BIC     #LEVUSD!NXTLEV,R3       ; NO ERROR RECOVERY IN PROGRESS
    103 004604  100653  000047                  MOV     R3,U.RCOV(R5)   ; SAVE
    104 004606  114001                  4$:     CLR     R1              ; IMMIDIATE CALL
    105 004607  003231                          BR      JMPRET          ; RETURN TO          R
    106                                         .DSABL  LSB
    107         004611                  AREA1   =       .+1
```

```
                                .SBTTL  ********** OVERLAY MODULE NEWOP - SET UP NEW OPERATION (COMMON CODE TOO)
                                        DMOVLY  NO,AREA1
004610  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
                                ;
000002                          NEWOP   =       SETUP+1
                                ;
                                ;       SET UP A NEW OPERATION
                                ;
                                        .ENABL  LSB
004611  104147                          MOV     (R4),R0         ; GET SUBUNIT PARAMETERS
004612                                  ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
004612  102207  040100                  BIT     #       NITW,R0         ; SEE IF BN'S ACCESSED SREQUENTIALLY
004614  054626                          BNE     2$              ; IF SO, BRANCH
004615  102207  000040                  BIT     #BEUSED,R0      ; SEE IF BEGIN/END SETS USED
004617  014623                          BEQ     1$              ; IF NOT, BRANCH
004620  104207  000003                  MOV     #RNDBE,R0       ; GET A RANDOM BLOCK NUMBER (BEGIN/END SETS)
004622  004636                          BR      4$              ; EXIT
004623  104207  000005          1$:     MOV     #RNDTG,R0       ; GET A RANDOM BLOCK NUMBER (TRACKS/GROUPS)
004625  004636                          BR      4$              ; BRANCH
004626  102207  000040          2$:     BIT     #BEUSED,R0      ; SEE IF BEGIN/END SETS USED
004630  014634                          BEQ     3$              ; IF NOT, BRANCH
004631  104207  000004                  MOV     #               ; GET A          AL BLOCK NUMBER (BEGIN/END SETS)
004633  004636                          BR      4$              ; EXIT (SHOULD BRANCH TO 7$) ********
004634  104207  000006          3$:     MOV     #               ; GET A          AL BLOCK NUMBER (TRACKS/GROUPS)
004636  114001                  4$:     CLR     R1              ; IMMIDATE CALL TO NEXT MODULE
004637  114002                          CLR     R2              ; NO ERRORS
004640  003231                          BR      JMPRET
                                        .DSABL  LSB
```

```
      1                                 .SBTTL  AFTOP  - AFTER THE SECTOR HAS BEEN DETERMINED, FIND OUT WHAT TO DO WITH IT
      2 004641                          AFTOP:
      3                                 ;
      4                                 ;       AFTER THE SECTORS TO BE READ/WRITTEN HAVE BEEN DETERMINED, SET UP THE
      5                                 ;       REST OF THE OPERATION
      6                                 ;
      7                                 .SBTTL  CLRUP  - CLEAR ALL PARAMETER BITS
      8                                 ;CLRUP
      9                                 ;
     10                                 ;       CLRUP CLEARS ALL NECESSARY FLAG BITS
     11                                 ;
     12 004641  104657  000046                  MOV     U.PARM(R5),R0   ; MOVE UNIT PARAMETERS TO R0
     13                                 ;       CLEAR ALL FLAGS BEFORE THE NEXT OPERATION
     14 004643  103207  004712                  BIC     #RBNBN!REVEC!WCHECK!DATCMP!REDWRT!NEWSUB,R0
     15 004645  100657  000046                  MOV     R0,U.PARM(R5)   ; SAVE UNIT PARAMETERS
```

```
                                        .SBTTL  RORW   - DETERMINE IF A READ OR A WRITE IS TO BE DONE
                                        ;RORW
                                        ;
                                        ;       RORW DETERMINES IF THE BLOCK SELECTED WILL BE READ OR WRITTEN
                                        ;
 004647  104142                         MOV     (R4),R2          ; GET SUBUNIT PARAMETERS
 004650                                 ASSUME  S.PARM,0         ; ASSUME THAT S.PARM IS ZERO
 004650  102202  042000                 BIT     #INITW!WONLY,R2  ; SEE IF INITIAL WRITE OR WRITE ONLY
 004652  054661                         BNE     1$               ; IF SO, BRANCH
 004653  102202  004000                 BIT     #RONLY,R2        ; SEE IF READ ONLY
 004655  054665                         BNE     2$               ; IF SO, BRANCH
 004656  024773                         CALL    RANDOM           ; GET RANDOM NUMBER
 004657  110601                         ROR     R1               ; ROTATE LOW BIT INTO CARRY
 004660  044665                         BCC     2$               ; IF LOW BIT ZERO, BRANCH
 004661  101207  000100          1$:    BIS     #REDWRT,R0       ; SET READ OR WRITE BIT (WRITE)
 004663  100657  000046                 MOV     R0,U.PARM(R5)    ; SAVE UNIT PARAMETERS
 004665                          2$:
```

```
      1                                 .SBTTL  PATTRN - IF WRITE, DETERMINE WHAT PATTERN IS TO BE WRITTEN
      2                                 ;PATTRN
      3                                 ;
      4                                 ;       FIND PATTERN TO USE (ONLY IF WRITE TO BE DONE)
      5                                 ;
      6 004665  102207  000100                   BIT     #REDWRT,R0      ; SEE IF WRITE TO BE DONE
      7 004667  014704                            BEQ     PATEXT          ; IF NOT, BRANCH
      8 004670  104641  000004                   MOV     S.PAT(R4),R1    ; GET PATTERN NUMBER
      9 004672  014676                            BEQ     RND0            ; IF PATTERN NOT SELECTED, BRANCH
     10 004673  103201  177760                   BIC     #LBLONB,R1      ; SPECIAL PATTERN USED, CLEAR HI BITS (MAP 16 TO 0)
     11 004675  004702                            BR      FIXPAT          ; IF PATTERN SELECTED, USE IT
     12 004676  024773                  RND0:    CALL    RANDOM          ; GET RANDOM PATTERN NUMBER
     13 004677  103201  177760                   BIC     #LBLONB,R1      ; CLEAR UNUSED BITS
     14 004701  014676                            BEQ     RND0            ; NO SUCH THING AS PATTERN 0, TRY AGAIN
     15 004702  100651  000014          FIXPAT:  MOV     R1,U.PAT(R5)    ; SAVE THE PATTERN NUMBER
     16 004704                          PATEXT:
```

```
                                        .SBTTL  WCHK   - IF WRITE, SEE IF A WRITE CHECK IS TO BE DONE
                                        ;WCHK
                                        ;
                                        ;       WCHK SETS THE WRITE CHECK BIT IF A WRITE CHECK IS TO BE DONE
                                        ;
 004704  102207  000100                         BIT     #REDWRT,R0      ; SEE IF WRITE IS TO BE DONE
 004706  014727                                 BEQ     2$              ; IF NOT, BRANCH
 004707  102202  000010                         BIT     #WCHECK,R2      ; SEE IF WRITE CHECK IS TO BE DONE
 004711  014727                                 BEQ     2$              ; IF NOT, BRANCH
 004712  102202  040000                         BIT     #INITW,R2       ; SEE IF INITIAL WRITE IS IN PROGRESS
 004714  054727                                 BNE     2$              ; IF SO, NO WRITE CHECK, SO BRANCH
 004715  102202  000004                         BIT     #WCHKAL,R2      ; SEE IF WRITE CHECK IS ALWAYS TO BE DONE
 004717  054723                                 BNE     1$              ; IF SO, BRANCH
 004720  024773                                 CALL    RANDOM          ; GET RANDOM NUMBER
 004721  110601                                 ROR     R1              ; 1/2 OF THE TIME WRITE CHECK
 004722  044727                                 BCC     2$              ; BRANCH (NO WRITE CHECK) IF LOWEST BIT CLEAR
 004723  101207  000010                 1$:     BIS     #WCHECK,R0      ; SET WRITE CHECK BIT
 004725  100657  000046                         MOV     R0,U.PARM(R5)   ; SAVE UNIT PARAMETERS
 004727                                 2$:
```

```
                                        .SBTTL  DCOMP  - IF READ OR WRITE W/WRITE CHECK, SEE IF DATA COMPARE IS TO BE DONE
                                        ;DCOMP
                                        ;
                                        ;       DCOMP SETS THE DATCMP BIT IF A DATA COMPARE IS TO BE DONE
                                        ;
 6 004727  102202  000002                       BIT     #DATCMP,R2      ; SEE IF DATA COMPARE REQUESTED
 7 004731  014752                               BEQ     DCEXT           ; IF NOT, BRANCH
 8 004732  102207  000100                       BIT     #REDWRT,R0      ; SEE IF READ IS TO BE DONE
 9 004734  014740                               BEQ     DCREAD          ; IF SO, BRANCH
10 004735  102207  000010                       BIT     #WCHECK,R0      ; SEE IF WRITE CHECK IS TO BE DONE
11 004737  014752                               BEQ     DCEXT           ; IF NOT, BRANCH
12 004740  102202  000001           DCREAD:     BIT     #DCMPAL,R2      ; SEE IF DATA COMPARE ALWAYS DONE
13 004742  054746                               BNE     DCMPYS          ; IF SO, BRANCH
14 004743  024773                               CALL    RANDOM          ; GET RANDOM NUMBER
15 004744  110601                               ROR     R1              ; SEE IF DATA COMPARE SHOULD BE DONE
16 004745  044752                               BCC     DCEXT           ; IF NOT, BRANCH
17 004746  101207  000002           DCMPYS:     BIS     #DATCMP,R0      ; SET DATA COMPARE BIT
18 004750  100657  000046                       MOV     R0,U.PARM(R5)   ; SAVE UNIT PARAMETERS
19 004752                           DCEXT:
```

```
UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE 68
DCOMP  - IF READ OR WRITE W/WRITE CHECK, SEE IF DATA COMPARE I

      1 004752  104657  000051                  MOV     U.MBN(R5),R0    ; GET LO LBN
      2 004754  100657  000053                  MOV     R0,U.CBN(R5)    ; SAVE
      3 004756  104657  000052                  MOV     U.MBN+1(R5),R0  ; GET HI LBN
      4 004760  100657  000054                  MOV     R0,U.CBN+1(R5)  ; SAVE
      5 004762  114002                          CLR     R2              ; NO ERRORS
      6 004763  100652  000024                  MOV     R2,U.CSEC(R5)   ; SECTORS R/W IS ZERO
      7 004765  100652  000021                  MOV     R2,U.NSEC(R5)   ; SECTORS R/W THIS PASS IS ZERO
      8 004767  114001                          CLR     R1              ; IMMIDATE CALL TO NEXT MODULE
      9 004770  104207  000001                  MOV     #SETUP,R0       ; NEXT MODULE IS SETUP
     10 004772  003231                          BR      JMPRET          ; RETURN TO          R
```

```
                                        .SBTTL  RANDOM - RANDOM NUMBER ROUTINE
004773                                  RANDOM:
                                        ;
                                        ;       RANDOM CALCULATES A RANDOM NUMBER AND RETURNS IT IN R1
                                        ;       RANGE 0 - 2^16-1
                                        ;
004773                                          PUSH    <R0,R2>         ; SAVE R0 AND R2
004773  100467                                                                          MOV R0,-(SP)
004774  100462                                                                          MOV R2,-(SP)
004775  104307  002224                          MOV     LOSEED,R0       ; MOVE LO ORDER SEED TO R0
004777  104301  002225                          MOV     HISEED,R1       ; MOVE HI SEED TO R1
005001  104202  000007                          MOV     #7,R2           ; MOVE LOOP COUNT TO R2
005003  110207                          1$:     ROL     R0              ; ROTATE LO ORDER NUMBER BY 1
005004  110201                                  ROL     R1              ; ROTATE HI OREDR NUMBER BY 1 (PROPOGATE CARRY)
005005  103207  000001                          BIC     #1,R0           ; CLEAR LOWER BIT
005007  117402                                  DEC     R2              ; DECREMENT COUNT
005010  055003                                  BNE     1$              ; IF COUNT INCOMPLETE, BRANCH
005011  105307  002224                          ADD     LOSEED,R0       ; ADD ORIGINAL SEED (X129)
005013  045015                                  BCC     2$              ; IF NO CARRY, BRANCH
005014  115401                                  INC     R1              ; PROPOGATE CARRY
005015  105301  002225                  2$:     ADD     HISEED,R1       ; ADD HISEED
005017  105207  001057                          ADD     #1057,R0        ; ADD LO CONSTANT
005021  045023                                  BCC     3$              ; IF NO CARRY, BRANCH
005022  115401                                  INC     R1              ; PROPOGATE CARRY
005023  105201  047401                  3$:     ADD     #47401,R1       ; ADD HI CONSTANT
005025  104070  002224                          MOV     R0,LOSEED       ; SAVE LO ORDER SEED
005027  104010  002225                          MOV     R1,HISEED       ; SAVE HI ORDER SEED
005031                                          POP     <R2,R0>         ; RESTORE R2 AND R0
005031  104262                                                                          MOV (SP)+,R2
005032  104267                                                                          MOV (SP)+,R0
005033  000000                                  RETURN
```

```
                                        .SBTTL  MASK    - FIND MASK FOR RANDOM NUMBER
005034                                  MASK:
                                        ;
                                        ;       MASK MASKS OUT HIGHER BITS OF A RANDOM NUMBER, SO THE PROB. OF THE
                                        ;       RANDOM NUMBER EXCEEDING THE MAXIMUM DESIRED IS LESSENED
                                        ;
005034                                          PUSH    R1              ; SAVE R1
005034  100461                                                                  MOV R1,-(SP)
005035  114001                                  CLR     R1              ; ZERO R1
005036  105011                          1$:     ADD     R1,R1           ; ROTATE R1 ONE BIT LEFT
005037  101201  000001                          BIS     #1,R1           ; TURN ON BIT 0
005041  106017                                  CMP     R1,R0           ; SEE IF R1 IS GREATER THAN R0
005042                                          BCS     1$              ; IF NOT, BRANCH
005042  045044                                                                  BCC     .+2
005043  005036                                                                  BR      1$
005044  104207  177777                          MOV     #-1,R0          ; SET UP FOR COMPLEMENT
005046  103017                                  BIC     R1,R0           ; COMPLEMENT R1
005047                                          POP     R1              ; RESTORE R1
005047  104261                                                                  MOV (SP)+,R1
005050  000000                                  RETURN                  ; RETURN TO RNDLBN
        005052                          AREA2   =       .+1
                                        .IF     LE,MAXADR-.
                                        MAXADR  =       .+1
                                        .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE RNDBE - GET NEXT RANDOM BEGIN/END SET SECTOR
                                        DMOVLY  RB,AREA2
005051  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
000003                          RNDBE   =       NEWOP+1
                                ;
                                ;       RNDBE GETS A RANDOM LBN NUMBER
                                ;
                                ;       GET A RANDOM BEGIN/END SET FROM THE SUBUNIT'S B/E SETS
                                ;
                                        .ENABL  LSB
005052  114007                          CLR     R0              ; INITILIZE COUNTER
005053  104203  000016                  MOV     #S.BESS+3,R3    ; R3 POINTS TO FIRST B/E SET
005055  105043                          ADD     R4,R3           ; R3 POINTS TO FIRST B/E SET
005056  104132                  CNTLOP: MOV     (R3),R2         ; MOVE EOL FLAG WORD TO R2
005057  075064                          BMI     COUNTD          ; IF END-OF-LIST, BRANCH
005060  105203  000004                  ADD     #4,R3           ; POINT TO NEXT EOL WORD
005062  115407                          INC     R0              ; INCREMENT COUNT
005063  005056                          BR      CNTLOP          ; IF NOT, TRY AGAIN
005064  115007                  COUNTD: TST     R0              ; SEE IF COUNT = 0
005065  015106                          BEQ     GOTOFS          ; IF SO, BRANCH
005066  024773                  TRYAGN: CALL    RANDOM          ; GET A RANDOM NUMBER
005067  103201  177774                  BIC     #177774,R1      ; MASK OUT ALL BUT 2 LOWEST BITS
005071  106207  000002                  CMP     #2,R0           ; TEST HOW COUNT RELATES TO 2
005073  075103                          BMI     GOTOBE          ; IF COUNT IS 3, BRANCH
005074  015100                          BEQ     THREBE          ; IF COUNT IS 2, BRANCH
005075  103201  177776                  BIC     #177776,R1      ; MASK OUT ALL BUT LOWEST BIT
005077  005103                          BR      GOTOBE          ; BRANCH
005100  106201  000002          THREBE: CMP     #2,R1           ; SEE IF RANDOM NUMBER OVER 2
005102  075066                          BMI     TRYAGN          ; IF SO, GET ANOTHER NUMBER
005103  104017                  GOTOBE: MOV     R1,R0           ; MOVE RANDOM NUMBER TO R0
005104  105077                          ADD     R0,R0           ; R0 X 2
005105  105077                          ADD     R0,R0           ; R0 X 2
005106  105207  000013          GOTOFS: ADD     #S.BESS,R0      ; POINT TO RANDOM BEGIN/END SET
005110  105047                          ADD     R4,R0           ; POINT TO RANDOM BEGIN/END SET
```

```
      1                                 ;
      2                                 ;       ONCE A RANDOM B/E SET HAS BEEN FOUND, FIND A RANDOM LBN
      3                                 ;       WITHIN THAT B/E SET
      4                                 ;
      5 005111                          ;       PUSH    <R4,R5>         ; SAVE R4 AND R5
        005111  100464                                                                  MOV R4,-(SP)
        005112  100465                                                                  MOV R5,-(SP)
      6 005113  104274                          MOV     (R0)+,R4        ; R4 GETS LO ORDER ENDING LBN
      7 005114  104275                          MOV     (R0)+,R5        ; R5 GETS HI ORDER ENDING LBN
      8 005115  107274                          SUB     (R0)+,R4        ; SUBTRACT LO ORDER STARTING LBN
      9 005116  045120                          BCC     NOBORO          ; IF NO BORROW, BRANCH
     10 005117  117405                          DEC     R5              ; ADJUST FOR BORROW
     11 005120  104171                NOBORO:   MOV     (R0),R1         ; GET HI WORD
     12 005121  103201  170000                  BIC     #170000,R1      ; STRIP OFF EOL FLAG (IF ANY)
     13 005123  107015                          SUB     R1,R5           ; SUBTRACT HI ORDER STARTING LBN
     14 005124  117407                          DEC     R0              ; POINT TO LO ORDER STARTING LBN
     15 005125                          ;       PUSH    R0              ; SAVE R0
        005125  100467                                                                  MOV R0,-(SP)
     16 005126  024773                  10$:    CALL    RANDOM          ; GET A RANDOM NUMBER
     17 005127  114003                          CLR     R3              ;
     18 005130  104301  002224                  MOV     LOSEED,R1       ; ORIGINAL SEED IN R1
     19 005132  115005                          TST     R5              ; SEE IF R5 IS ZERO
     20 005133  015145                          BEQ     11$             ; IF SO, BRANCH
     21 005134  104057                          MOV     R5,R0           ; ELSE, STORE IN R0
     22 005135  025034                          CALL    MASK            ; COMPUTE RANDOM NUMBER MASK
     23 005136  104303  002225                  MOV     HISEED,R3       ; MOVE RANDOM NUMBER INTO R3
     24 005140  103073                          BIC     R0,R3           ; CLEAR WITH MASK
     25 005141  106053                          CMP     R5,R3           ; CMP MAXIMUM RANDOM NUM TO RANDOM NUM
     26 005142                          ;       BCS     10$             ; IF TOO LARGE, DO AGAIN
        005142  045144                                                                  BCC     .+2
        005143  005126                                                                  BR      10$
     27 005144  055153                          BNE     CLCLBN          ; IF NOT THE SAME, BRANCH
     28 005145  104047                  11$:    MOV     R4,R0           ; MOVE R4 TO R0 TO COMPUTE MASK
     29 005146  025034                          CALL    MASK            ; COMPUTE MASK
     30 005147  103071                          BIC     R0,R1           ; MASK OUT HIGH BITS
     31 005150  106041                          CMP     R4,R1           ; SEE IF MAX RND NUM < RANDOM NUMBER
     32 005151                          ;       BCS     10$             ; IF SO, DO AGAIN
        005151  045153                                                                  BCC     .+2
        005152  005126                                                                  BR      10$
     33 005153  104012                  CLCLBN: MOV     R1,R2           ; MOVE RANDOM NUMBER TO R2
     34 005154                          ;       POP     R0              ; RESTORE R0
        005154  104267                                                                  MOV (SP)+,R0
     35 005155  105272                          ADD     (R0)+,R2        ; ADD LO ORDER STARTING BLOCK TO RANDOM NUM
     36 005156  045160                          BCC     3$              ; IF NO CARRY, BRANCH
     37 005157  115403                          INC     R3              ; ADJUST HIGH ORDER FOR CARRY
     38 005160  104171                  3$:     MOV     (R0),R1         ; MOVE HI ORDER LBN TO R1
     39 005161  103201  170000                  BIC     #170000,R1      ; STRIP OFF EOL FLAG IF ANY
     40 005163  105013                          ADD     R1,R3           ; ADD HI ORDER LBN
     41 005164                          ;       POP     <R5,R4>         ; RESTORE R5, R4
        005164  104265                                                                  MOV (SP)+,R5
        005165  104264                                                                  MOV (SP)+,R4
     42 005166  107207  000003                  SUB     #3,R0           ; R0 POINTS TO ENDING LBN
     43 005170  104271                          MOV     (R0)+,R1        ; R1 HAS LO ORDER ENDING LBN
     44 005171  104177                          MOV     (R0),R0         ; R0 HAS HI ORDER ENDING LBN
     45 005172  107021                          SUB     R2,R1           ; SUBTRACT LO LBN FROM LO ENDING LBN
     46 005173  075200                          BMI     4$              ; IF RESULT IS NEGATIVE, BRANCH
     47 005174  045176                          BCC     1$              ; IF NO CARRY, BRANCH
```

```
     48 005175  117407                      DEC     R0              ; DECREMENT HI ENDING LBN
     49 005176  107037                1$:   SUB     R3,R0           ; SUBTRACT HI LBN FROM HI ENDING LBN
     50 005177  015202                      BEQ     2$              ; IF HI WORD IS ZERO, BRANCH
     51 005200  104201  077776        4$:   MOV     #77776,R1       ; MOVE MAXIMUM COUNT TO R1
     52 005202  115401                2$:   INC     R1              ; INCREMENT MAXIMUM SECTOR COUNT
     53 005203  104017                      MOV     R1,R0           ; COPY TO R0 FOR MAXMUM
     54
     55 005204                              PUSH    R3              ; SAVE R3
        005204  100463                                                      MOV R3,-(SP)
     56 005205  024773                      CALL    RANDOM          ; GET A RANDOM NUMBER OF SECTORS TO READ/WRITE
     57 005206  103201  177774              BIC     #MAXMSK,R1      ; SET READ/WRITE LIMIT
     58 005210  115401                      INC     R1              ; MAKE MAXIMUM OF LIMIT+1
     59 005211  106071                      CMP     R0,R1           ; SEE IF RANDOM NUMBER EXCEEDS POSSIBLE
     60 005212  075214                      BMI     6$              ; IF SO, BRANCH
     61 005213  104017                      MOV     R1,R0           ; ONLY READ/WRITE THE RANDOM NUMBER OF SECTORS
     62 005214  100657  000023        6$:   MOV     R0,U.TSEC(R5)   ; SAVE IN TSEC
     63 005216                              POP     R3              ; RESTORE R3
        005216  104263                                                      MOV (SP)+,R3
     64 005217  104207  000051              MOV     #U.MBN,R0       ; R0 POINTS TO LBN AREA
     65 005221  105057                      ADD     R5,R0           ; R0 POINTS TO LBN AREA
     66 005222  100272                      MOV     R2,(R0)+        ; MOVE LO ORDER TO LBN AREA
     67 005223  100173                      MOV     R3,(R0)         ; MOVE HI ORDER TO LBN AREA
     68 005224  104207  004641              MOV     #AFTOP,R0       ; NEXT MODULE IS AFTOP
     69 005226  114001                      CLR     R1              ; IMMIDATE CALL
     70 005227  114002                      CLR     R2              ; NO ERRORS
     71 005230  003231                      BR      JMPRET
     72                                     .DSABL  LSB
     73                                     .IF     LE,MAXADR-.
     74                             MAXADR  =       .+1
     75                                     .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE        GET NEXT        AL BEGIN/END SET SECTOR
005231                                  DMOVLY  SB,AREA2
005231  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
000004                                  RNDBE+1
                                ;
                                ;               NDS THE NEXT            AL LBN
                                ;
                                .ENABL  LSB
005052  104657  000046                  MOV     U.PARM(R5),R0   ; MOVE UNIT PARAMETERS TO R0
005054  102207  010000                  BIT     #DIREC,R0       ; TEST DIRECTION
005056  015163                          BEQ     UP              ; IF          AL UP, BRANCH
005057  115000  002217                  TST     SCR1            ; SEE IF NEW SUBUNIT
005061  015104                          BEQ     5$              ; IF NOT, BRANCH
005062  104201  000013                  MOV     #S.BESS,R1      ; R0 WILL POINT TO BEGIN/END SETS
005064  105041                          ADD     R4,R1           ; R0 POINTS TO BEGIN/END SETS
005065  104617  000003          6$:     MOV     3(R1),R0        ; SEE IF END-OF-LIST
005067  075073                          BMI     8$              ; IF SO, BRANCH
005070  105201  000004                  ADD     #4,R1           ; POINT TO NEXT BEGIN/END SET
005072  005065                          BR      6$              ; LOOP
005073  104202  000051          8$:     MOV     #U.MBN,R2       ; R2 WILL POINT TO MASTER BN
005075  105052                          ADD     R5,R2           ; R2 POINTS TO MASTER BN
005076  104217                          MOV     (R1)+,R0        ; GET LO ORDER ENDING SET
005077  100127-                         MOV     R0,(R2)         ; SAVE
005100  104217                          MOV     (R1)+,R0        ; GET HI ORDER ENDING SET
005101  100627  000001                  MOV     R0,1(R2)        ; SAVE
005103  005123                          BR      7$              ; BRANCH
005104  025275                  5$:     CALL    FNDBES          ; FIND BEGIN/END SET THAT LBN IS IN
005105  105201  000002                  ADD     #2,R1           ; POINT TO BEGIN SET
005107  021765                          CALL    CMP2            ; SEE IF LBN = BEGINNING LBN
005110  015143                          BEQ     BESDWN          ; IF SO, BRANCH
005111  104127                          MOV     (R2),R0         ; MOVE LOW ORDER WORD TO R0
005112  107207  000001                  SUB     #1,R0           ; DECREMENT R0
005114  100127                          MOV     R0,(R2)         ; SAVE R0
005115  045123                          BCC     7$              ; IF NO BORROW, BRANCH
005116  104627  000001                  MOV     1(R2),R0        ; MOVE HIGH ORDER WORD TO R0
005120  117407                          DEC     R0              ; DECREMENT R0
005121  100627  000001                  MOV     R0,1(R2)        ; SAVE R0
005123  104227                  7$:     MOV     (R2)+,R0        ; MOVE LO ORDER BN TO R0
005124  104223                          MOV     (R2)+,R3        ; MOVE HI ORDER BN TO R3
005125  107217                          SUB     (R1)+,R0        ; SUBTRACT BEGIN BN FROM BN
005126  075136                          BMI     4$              ; IF RESULT IS NEGATIVE, BRANCH
005127  045131                          BCC     1$              ; IF NO BORROW, BRANCH
005130  117403                          DEC     R3              ; PROPOGATE BORROW
005131  104111                  1$:     MOV     (R1),R1         ; GET BEGINNING BN
005132  103201  170000                  BIC     #^CHBHINB,R1    ; CLEAR EOL FLAG, IF ANY
005134  107013                          SUB     R1,R3           ; SUBTRACT HI ORDER BN FROM HI BN
005135  015140                          BEQ     2$              ; IF EQUAL, BRANCH
005136  104207  077776          4$:     MOV     #77776,R0       ; MOVE MAXIMUM COUNT TO R0
005140  115407                  2$:     INC     R0              ; MAKE SUBTRACTION INCLUSIVE
005141  025314                  3$:     CALL    MAXMUM          ; SET MAXIMUM COUNT (U.TSEC AND U.MSEC)
```

```
                                                        BR      NLBEXT          ; BRANCH
                                                        .DSABL  LSB
                                                BESDWN: CALL    PREVBE          ; FIND PREVIOUS B/E SET OR UNIT
                                                NLBEXT: MOV     U.TSEC(R5),R1   ; GET TOTAL NUMBER OF SECTORES TO READ/WRITE
                                                        DEC     R1              ; DECREMENT SECTOR COUNT
                                                        MOV     U.MBN(R5),R3    ; GET LO ORDER MASTER BN
                                                        SUB     R1,R3           ; SUBTRACT SECTORS TO READ/WRITE
                                                        MOV     R3,U.MBN(R5)    ; SAVE
                                                        BCC     1$              ; IF NO CARRY, BRANCH
                                                        MOV     U.MBN+1(R5),R3  ; GET HI ORDER MASTER BN
                                                        DEC     R3              ; PROPOGATE CARRY
                                                        MOV     R3,U.MBN+1(R5)  ; SAVE
                                                1$:     BR      BEEXT           ; BRANCH
                                                UP:     TST     SCR1            ; SEE IF FIRST TIME ON THIS SUBUNIT
                                                        BEQ     5$              ; IF NOT, BRANCH
                                                        MOV     #S.BESS,R1      ; R0 WILL POINT TO BEGIN/END SETS
                                                        ADD     R4,R1           ; R0 POINTS TO BEGIN/END SETS
                                                        MOV     #U.MBN,R2       ; R2 WILL POINT TO MASTER BN
                                                        ADD     R5,R2           ; R2 POINTS TO MASTER E I
                                                        MOV     2(R1),R0        ; GET LO ORDER STARTING SET
                                                        MOV     R0,(R2)         ; SAVE
                                                        MOV     3(R1),R0        ; GET HI ORDER STARTING SET
                                                        BIC     #^CHBHINB,R0    ; CLEAR EOL FLAG IF ANY
                                                        MOV     R0,1(R2)        ; SAVE
                                                        BR      7$              ; BRANCH
                                                5$:
                                                .SBTTL  UPLBN  - IF GOING UP, UPDATE CURRENT BN TO LAST BN READ/WRITTEN
                                                ;UPLBN
                                                ;
                                                ;       UPLBN UPDATES U.MBN TO POINT TO THE LAST LBN READ/WRITTEN
                                                ;       (INCREASING LBN)
                                                ;
005206                                          PUSH    <R0,R2>         ; SAVE R0,R2
005206  100467                                                                  MOV R0,-(SP)
005207  100462                                                                  MOV R2,-(SP)
005210  104202  000051                          MOV     #U.MBN,R2       ; R2 POINTS TO LAST LBN
005212  105052                                  ADD     R5,R2           ; R2 POINTS TO LAST LBN
005213  104657  000023                          MOV     U.TSEC(R5),R0   ; GET NUMBER OF SECTORS WRITTEN
005215  117407                                  DEC     R0              ; SUBTRACT ONE
005216  105127                                  ADD     (R2),R0         ; ADD LOW ORDER LBN TO R0
005217  100227                                  MOV     R0,(R2)+        ; SAVE LOW ORDER WORD, POINT TO HIGH ORDER
005220  045224                                  BCC     8$              ; IF NO CARRY, BRANCH
005221  104127                                  MOV     (R2),R0         ; GET HIGH ORDER WORD
005222  115407                                  INC     R0              ; INCREMENT
005223  100127                                  MOV     R0,(R2)         ; SAVE HIGH ORDER WORD
005224                                  8$:     POP     <R2,R0>         ; RESTORE R0,R2
005224  104262                                                                  MOV (SP)+,R2
005225  104267                                                                  MOV (SP)+,R0
005226  025275                                  CALL    FNDBES          ; FIND BEGIN/END SET THAT LBN IS IN
005227  015257                                  BEQ     BESUP           ; IF LBN = ENDING LBN, BRANCH
005230  104127                                  MOV     (R2),R0         ; MOVE LOW ORDER WORD TO R0
005231  105207  000001                          ADD     #1,R0           ; INCREMENT R0
005233  100127                                  MOV     R0,(R2)         ; SAVE R0
005234  045242                                  BCC     7$              ; IF NO CARRY, BRANCH
005235  104627  000001                          MOV     1(R2),R0        ; MOVE HIGH ORDER WORD TO R0
005237  115407                                  INC     R0              ; INCREMENT R0
005240  100627  000001                          MOV     R0,1(R2)        ; SAVE R0
```

```
110 005242  104217                  7$:     MOV     (R1)+,R0        ; R0 HAS LO ENDING LBN
111 005243  104113                          MOV     (R1),R3         ; R3 HAS HI ENDING LBN
112 005244  107227                          SUB     (R2)+,R0        ; SUBTRACT LO LBN FROM LO ENDING LBN
113 005245  075252                          BMI     4$              ; IF RESULT IS NEGATIVE, BRANCH
114 005246  045250                          BCC     1$              ; IF NO CARRY, BRANCH
115 005247  117403                          DEC     R3              ; PROPOGATE CARRY
116 005250  107123                  1$:     SUB     (R2),R3         ; SUBTRACT HI LBN FROM ENDING LBN
117 005251  015254                          BEQ     2$              ; IF ZERO, BRANCH
118 005252  104207  077776          4$:     MOV     #77776,R0       ; MOVE MAXIMUM COUNT TO R0
119 005254  115407                  2$:     INC     R0              ; MAKE SUBTRACTION INCLUSIVE
120 005255  025314                  3$:     CALL    MAXMUM          ; SET MAXIMUM COUNT (U.TSEC AND U.MSEC)
121 005256  005260                          BR      BEEXT           ; BRANCH
122 005257  025370                  BESUP:  CALL    NEXTBE          ; FIND NEXT B/E SET OR UNIT
123 005260  104657  000046          BEEXT:  MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
124 005262  102207  004000                  BIT     #NEWSUB,R0      ; SEE IF NEW SUBUNIT
125 005264  015270                          BEQ     1$              ; IF NOT, BRANCH
126 005265  104207  000002                  MOV     #NEWOP,R0       ; NEWOP NEXT MODULE
127 005267  005272                          BR      2$              ; BRANCH
128 005270  104207  004641          1$:     MOV     #AFTOP,R0       ; NEXT MODULE IS AFTOP
129 005272  114001                  2$:     CLR     R1              ; IMMIDATE CALL
130 005273  114002                          CLR     R2              ; NO ERRORS
131 005274  003231                          BR      JMPRET          ; RETURN TO         R
```

```
      1                                          .SBTTL  FNDBES - FIND THE BEGIN/END SET CURRENT BN RESIDES IN
      2 005275                                   FNDBES:
      3                                          ;
      4                                          ;       FNDBES FINDS THE BEGIN/END SET THAT THE CURRENT LBN RESIDES IN
      5                                          ;
      6 005275  104203  177777                           MOV     #-1,R3          ; START COUNTER (ADJUSTED)
      7 005277  104201  000007                           MOV     #S.BESS-4,R1    ; POINT TO B/E SETS
      8 005301  105041                                   ADD     R4,R1           ; POINT TO B/E SETS
      9 005302  104202  000051                           MOV     #U.MBN,R2       ; POINT TO LBN
     10 005304  105052                                   ADD     R5,R2           ; POINT TO LBN
     11 005305  105201  000004                 FNDLOP:   ADD     #4,R1           ; INCREMENT B/E SET POINTER
     12 005307  115403                                   INC     R3              ; INCREMENT COUNT
     13 005310  021765                                   CALL    CMP2            ; COMPARE LBN WITH ENDING LBN
     14 005311  015313                                   BEQ     OUT0            ; IF = TO, BRANCH
     15 005312  045305                                   BCC     FNDLOP          ; IF LBN > ENDING LBN, BRANCH
     16 005313  000000                 OUT0:     RETURN                          ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  MAXMUM - FIND HOW MANY SECTORS TO READ/WRITE
005314                                  MAXMUM:
                                        ;
                                        ;       MAXMUM WILL WRITE THE MAXIMUM NUMBER OF SECTORS IF IN INITIAL WRITE,
                                        ;       OTHERWISE IT WILL R/W ONE TRACK
                                        ;
005314  115000  002223                        TST     M.PARM          ; SEE IF INITIAL WRITE IS IN PROGRESS
005316  075331                                BMI     1$              ; IF SO, BRANCH
005317                                        ASSUME  IWIPRG,100000   ; ASSUME IWIPRG IS SIGN BIT
005317  115000  002217                        TST     SCR1            ; SEE IF FIRST TIME ON THIS UNIT
005321  015324                                BEQ     2$              ; IF NOT, BRANCH
005322  025442                                CALL    NXTTRK          ; ALIGN WITH NEXT TRACK
005323  005333                                BR      3$              ; EXIT
005324  106647  000006                2$:     CMP     S.TRKL(R4),R0   ; SEE IF TRACK GREATER THAN MAX
005326  035331                                BPL     1$              ; IF LESS THAN TRACK LENGTH, BRANCH
005327  104647  000006                        MOV     S.TRKL(R4),R0   ; R/W ONE TRACK
005331  100657  000023                1$:     MOV     R0,U.TSEC(R5)   ; SAVE NUMBER OF SECTORS TO R/W
005333  000000                        3$:     RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  PREVBE - MOVE TO PREVIOUS BEGIN/END SET
005334                                  PREVBE:
                                        ;
                                        ;       PREVBE IS CALLED IF THE LBN'S ARE DECREASING, AND THE LAST LBN
                                        ;       WAS THE FIRST LBN WITHIN A B/E SET.  PREVBE FINDS THE PREVIOUS
                                        ;       B/E SET, OR, IF IT IS THE LAST ON THE SUBUNIT, FINDS THE LAST B/E
                                        ;       SET OF THE PREVIOUS UNIT.  IF IT IS THE FIRST BE SET OF THE FIRST
                                        ;       UNIT, THE DIRECTION OF THE LBN'S ARE REVERSED (THEY BEGIN GOING UP
                                        ;       THE UNIT)
                                        ;
005334  115003                          TST     R3              ; SEE IF ALLREADY ON 1ST B/E SET
005335  055345                          BNE     5$              ; IF NOT, BRANCH
005336  104657  000046                  MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
005340  101207  004000                  BIS     #NEWSUB,R0      ; MARK AS GOING ON TO THE NEXT UNIT
005342  100657  000046                  MOV     R0,U.PARM(R5)   ; SAVE
005344  005367                          BR      6$              ; EXIT
005345  107201  000006          5$:     SUB     #6,R1           ; POINT TO PREVIOUS ENDING SET
005347  025427                          CALL    MOV2            ; MOVE ENDING SET TO LBN
005350  104217                          MOV     (R1)+,R0        ; R0 HAS LO ENDING LBN
005351  104212                          MOV     (R1)+,R2        ; R2 HAS HI ENDING LBN
005352  107217                          SUB     (R1)+,R0        ; SUBTRACT LO BEGINNING LBN FROM LO ENDING
005353  075363                          BMI     4$              ; IF RESULT IS NEGATIVE, BRANCH
005354  045356                          BCC     1$              ; IF NO CARRY, BRANCH
005355  117402                          DEC     R2              ; PROPOGATE CARRY
005356  104111                  1$:     MOV     (R1),R1         ; GET BEGINNING BN
005357  103201  170000                  BIC     #^CHBHINB,R1    ; CLEAR EOL FLAG, IF ANY
005361  107012                          SUB     R1,R2           ; SUBTRACT HI ORDER BN FROM HI BN
005362  015365                          BEQ     2$              ; IF ZERO, BRANCH
005363  104207  077776          4$:     MOV     #77776,R0       ; MOVE MAXIMUM COUNT TO R0
005365  115407                  2$:     INC     R0              ; TO MAKE IT AN INCLUSIVE SUBTRACT
005366  025442                  3$:     CALL    NXTTRK          ; SET MAXIMUM COUNT (U.TSEC AND U.MSEC)
005367  000000                  6$:     RETURN                  ; RETURN TO CALLING PROGRAM
```

NEXTBE - MOVE TO NEXT BEGIN/END SET

```
                                        .SBTTL  NEXTBE - MOVE TO NEXT BEGIN/END SET
005370                                  NEXTBE:
                                        ;
                                        ;       NEXTBE IS CALLED WHEN THE CURRENT LBN IS THE LAST OF ITS B/E SET.
                                        ;       NEXTBE GOES TO THE NEXT B/E SET FOR THE NEXT LBN.  IF THE LBN IS
                                        ;       THE LAST ON THE SUBUNIT, IT GOES TO THE FIRST B/E SET OF THE NEXT
                                        ;       UNIT.  IF IT IS THE LAST B/E SET ON THE LAST UNIT, THE DIRECTION IS
                                        ;       CHANGED (DOWN), AND THE INITIAL WRITE BIT IS CLEARED IN CASE AN
                                        ;       INITIAL WRITE IS IN  PROGRESS
                                        ;
005370  104617  000003                          MOV     3(R1),R0        ; SEE IF THIS IS LAST B/E SET ON THIS UNIT
005372  035402                                  BPL     1$              ; IF SO, BRANCH
005373  104657  000046                          MOV     U.PARM(R5),R0   ; GET SUBUNIT PARAMETERS
005375  101207  004000                          BIS     #NEWSUB,R0      ; MARK AS GOING ON TO A NEW SUBUNIT
005377  100657  000046                          MOV     R0,U.PARM(R5)   ; SAVE
005401  005426                                  BR      7$              ; EXIT
005402  105201  000006                  1$:     ADD     #6,R1           ; POINT TO NEXT BEGINNING LBN
005404  025427                                  CALL    MOV2            ; MOVE BEGINNING LBN TO LBN
005405  107201  000002                          SUB     #2,R1           ; R1 POINTS TO END SET
005407  104217                                  MOV     (R1)+,R0        ; R0 HAS LO ENDING LBN
005410  104213                                  MOV     (R1)+,R3        ; R3 HAS HI ENDING SET
005411  107217                                  SUB     (R1)+,R0        ; SUBTRACT BEGINNING LBN
005412  075422                                  BMI     5$              ; IF RESULT IS NEGATIVE, BRANCH
005413  045415                                  BCC     2$              ; IF NO CARRY, BRANCH
005414  117403                                  DEC     R3              ; PROPOGATE CARRY
005415  104111                  2$:     MOV     (R1),R1         ; GET BEGINNING BN
005416  103201  170000                          BIC     #^CHBHINB,R1    ; CLEAR EOL FLAG, IF ANY
005420  107013                                  SUB     R1,R3           ; SUBTRACT HI ORDER BN FROM HI BN
005421  015424                                  BEQ     3$              ; IF ZERO, BRANCH
005422  104207  077776          5$:     MOV     #77776,R0       ; MOVE MAXIMUM COUNT TO R0
005424  115407                  3$:     INC     R0              ; INCREMENT R0 (INCLUSIVE SUBTRACT)
005425  025442                  4$:     CALL    NXTTRK          ; SET MAXIMUM COUNT (U.TSEC AND U.MSEC)
005426  000000                  7$:     RETURN                  ; RETURN TO CALLING PROGRAM
```

MOV2 - 28 BIT MOVE

```
                                .SBTTL  MOV2   - 28 BIT MOVE
005427                          MOV2:
                                ;
                                ;       MOV2 COPIES A 28 BIT NUMBER POINTED TO BY R1 TO TWO WORDS POINTED
                                ;       TO BY R2.  BITS 31-28 ARE CLEARED BEFORE COPYING
                                ;
005427                                  PUSH    R0              ; SAVE R0
005427  100467                                                                  MOV R0,-(SP)
005430  104117                         MOV     (R1),R0         ; MOVE SOURCE LOW ORDER WORD TO R0
005431  100127                         MOV     R0,(R2)         ; MOVE R0 TO DESTINATION LOW ORDER WORD
005432  104617  000001                 MOV     1(R1),R0        ; MOVE SOURCE HIGH ORDER WORD TO R0
005434  103207  170000                 BIC     #^CHBHINB,R0    ; STRIP OFF UNUSED BITS
005436  100627  000001                 MOV     R0,1(R2)        ; MOVE R0 TO DESTINATION HIGH ORDER WORD
005440                                  POP     R0              ; RESTORE R0
005440  104267                                                                  MOV (SP)+,R0
005441  000000                         RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                .SBTTL  NXTTRK - FIND HOW MANY SECTORS TO R/W TO ALIGN WITH NEXT TRACK BOUNDRY
005442                          NXTTRK:
                                ;
                                ;       COMPUTE HOW MANY SECTORS TO READ/WRITE TO GET ON A TRACK BOUNDRY
                                ;       THIS WILL ALLOW         AL READS AND WRITES TO BE PERFORMED
                                ;       WITHOUT ANY 'BACKWARDS' SEEKING (EXCEPT IN THE CASE OF REVECTORS)
                                ;
005442                                  PUSH    R0              ; SAVE R0
005442  100467                                                                 MOV R0,-(SP)
005443  115000  002223                 TST     M.PARM          ; SEE IF INITIAL WRITE
005445  075562                         BMI     7$              ; IF SO, EXIT (WRITE MAXIMUM POSSIBLE)
005446                                  ASSUME  IWIPRG,100000   ; ASSUME IWIPRG IS SIGN BIT
005446  104651  000051                 MOV     U.MBN(R5),R1    ; GET LO ORDER SECTOR TO R/W
005450  104010  002175                 MOV     R1,CURBN        ; MOVE TO CALC AREA
005452  104651  000052                 MOV     U.MBN+1(R5),R1  ; GET HI ORDER SECTOR TO R/W
005454  104010  002176                 MOV     R1,CURBN+1      ; MOVE TO CALC AREA
005456  114007                         CLR     R0              ; TELL CALC TO SETUP ALL REQUIRED PARAMETERS
005457  022027                         CALL    CALC            ; CALC TRACK THAT SECTOR IS ON
005460  104207  005604                 MOV     #CYLSCR,R0      ; R0 POINTS TO SCRATCH SPACE
005462  104201  002201                 MOV     #CYL,R1         ; R1 POINTS TO CYL, GRP AND TRACK
005464  104202  000004                 MOV     #4,R2           ; MOVE 4 WORDS
005466  104213                 8$:     MOV     (R1)+,R3        ; GET WORD
005467  100273                         MOV     R3,(R0)+        ; SAVE
005470  117402                         DEC     R2              ; DECREMENT COUNT
005471  055466                         BNE     8$              ; IF INCOMPLETE, BRANCH
005472  104207  000020                 MOV     #20,R0          ; START INCREMENT BY 10 (16 DECIMAL)
005474  104651  000046         1$:     MOV     U.PARM(R5),R1   ; GET UNIT PARAMETERS
005476  025565                         CALL    ID              ; INCREMENT OR DECREMENT (DEPENDING ON DIRECTION)
005477                                  PUSH    R0              ; SAVE R0
005477  100467                                                                 MOV R0,-(SP)
005500  022027                         CALL    CALC            ; CALCULATE NEW CYL/GRP/TRK
005501  104207  005604                 MOV     #CYLSCR,R0      ; R0 POINTS TO SCRATCH SPACE
005503  104201  002201                 MOV     #CYL,R1         ; R1 POINTS TO CYL, GRP AND TRACK
005505  104202  000004                 MOV     #4,R2           ; MOVE 4 WORDS
005507  104213                 9$:     MOV     (R1)+,R3        ; GET WORD
005510  106273                         CMP     (R0)+,R3        ; COMPARE
005511  055521                         BNE     2$              ; IF CHANGE, BRANCH
005512  117402                         DEC     R2              ; DECREMENT COUNT
005513  055507                         BNE     9$              ; IF INCOMPLETE, BRANCH
005514                                  POP     R0              ; RESTORE STEPSIZE
005514  104267                                                                 MOV (SP)+,R0
005515  106207  000020                 CMP     #20,R0          ; SEE IF ORIGINAL STEPSIZE
005517  015474                         BEQ     1$              ; IF SO, BRANCH
005520  005532                         BR      3$              ; BRANCH
005521                          2$:     POP     R0              ; RESTORE STEPSIZE
005521  104267                                                                 MOV (SP)+,R0
005522  106207  000001                 CMP     #1,R0           ; SEE IF LAST STEPSIZE
005524  015540                         BEQ     4$              ; FOUND FIRST SECTOR ON ANOTHER TRACK
005525  104201  177777                 MOV     #-1,R1          ; SET UP FOR COMPLEMENT
005527  040651  000046                 BIC     U.PARM(R5),R1   ; COMPLEMENT
005531  025565                         CALL    ID
005532  110607                 3$:     RORB    R0              ; ROTATE TO HALVE STEPSIZE
005533  103207  177760                 BIC     #LBLONB,R0      ; CLEAR UNUSED BITS
005535  055474                         BNE     1$              ; IF NON-ZERO, LOOP
005536  115407                         INC     R0              ; MAKE R3 1
005537  005474                         BR      1$              ; LOOP
005540  104653  000046         4$:     MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
```

NXTTRK - FIND HOW MANY SECTORS TO R/W TO ALIGN WITH NEXT TRACK

```
 54 005542  102203  010000                BIT     #DIREC,R3       ; FIND WHAT DIRECTION YOU'RE GOING IN
 55 005544  015552                        BEQ     5$              ; IF UP, BRANCH
 56 005545  104653  000051                MOV     U.MBN(R5),R3    ; GET LO ORDER STARTING BN
 57 005547  107303  002175                SUB     CURBN,R3        ; SUBTRACT ENDING
 58 005551  005556                        BR      6$              ; GO WITH COMPUTED NUMBER
 59 005552  104303  002175          5$:   MOV     CURBN,R3        ; GET LO ORDER ENDING BN
 60 005554  107653  000051                SUB     U.MBN(R5),R3    ; SUBTRACT STARTING BN
 61 005556                          6$:   POP     R0              ; GET MAX SECTORS THAT CAN BE R/W
    005556  104267                                                                  MOV (SP)+,R0
 62 005557  106073                        CMP     R0,R3           ; COMPARE
 63 005560  075562                        BMI     7$              ; IF CAN WRITE MORE THAN POSSIBLE, BRANCH
 64 005561  104037                        MOV     R3,R0           ; SETUP TO WRITE TO EOT
 65 005562  100657  000023          7$:   MOV     R0,U.TSEC(R5)   ; SAVE IN SECTORS TO R/W
 66 005564  000000                        RETURN                  ; RETURN TO CALLING PROGRAM
 67
 68
 69 005565                          ID:
 70 005565  102201  010000                BIT     #DIREC,R1       ; SEE IF GOING UP OR DOWN
 71 005567  015576                        BEQ     1$              ; IF UP, BRANCH
 72 005570  107070  002175                SUB     R0,CURBN        ; DECREMENT BN
 73 005572  045603                        BCC     2$              ; IF NO BORROW, EXIT
 74 005573  117400  002176                DEC     CURBN+1         ; PROPOGATE BORROW
 75 005575  005603                        BR      2$              ; EXIT
 76 005576  105070  002175          1$:   ADD     R0,CURBN        ; INCREMENT BN
 77 005600  045603                        BCC     2$              ; IF NO CARRY, BRANCH
 78 005601  115400  002176                INC     CURBN+1         ; PROPOGATE CARRY
 79 005603  000000                  2$:   RETURN
 80
 81
 82 005604  000000          CYLSCR: .WORD   0               ; SCRATCH AREA FOR STORING CYL, GRP AND TRK
 83 005605  000000                  .WORD   0
 84 005606  000000                  .WORD   0
 85 005607  000000                  .WORD   0
 86                                 .IF     LE,MAXADR-.
 87                          MAXADR =       .+1
 88                                 .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE RNDTG - RANDOMLY GET NEXT TRACK/GROUP SECTOR
005610                                  DMOVLY  RT,AREA2
005610  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
                                ;
000005                          RNDTG   =
                                ;
                                ;       COME UP WITH A RANDOM BN TO TEST USING THE TRACK/GROUP SETS
                                ;
                                ;       FIRST COME UP WITH A RANDOM COUNT <= MAXIMUM COUNT
                                ;
                                        .ENABL  LSB
005052  024773                  1$:    CALL    RANDOM          ; GET RANDOM NUMBER
005053  114002                         CLR     R2              ; HI ORDE WORD IS ZERO
005054  104301  002224                 MOV     LOSEED,R1       ; LOSEED IN R1
005056  104647  000014                 MOV     S.MCNT+1(R4),R0 ; GET MAXIMUM HI ORDER WORD
005060  015072                         BEQ     3$              ; IF ZERO, BRANCH
005061  025034                         CALL    MASK            ; CREATE MASK FOR RANDOM NUMBER
005062  104302  002225                 MOV     HISEED,R2       ; PUT NUMBER IN R2
005064  103072                         BIC     R0,R2           ; STRIP OFF UNUSED BITS
005065  106642  000014                 CMP     S.MCNT+1(R4),R2 ; SEE IF MAX EXCEEDED
005067                                 BCS     1$              ; IF SO, TRY AGAIN
005067  045071                                                                 BCC     .+2
005070  005052                                                                 BR      1$
005071  055102                         BNE     5$              ; IF NOT EQUAL, EXIT
005072  104647  000013          3$:    MOV     S.MCNT(R4),R0   ; GET LO ORDER MAX
005074  025034                         CALL    MASK            ; GET RANDOM MASK
005075  103071                         BIC     R0,R1           ; STRIP OFF BITS
005076  106641  000013                 CMP     S.MCNT(R4),R1   ; CHECK WITH MAX
005100                                 BCS     1$              ; IF GREATER THAN MAX, DO AGAIN
005100  045102                                                                 BCC     .+2
005101  005052                                                                 BR      1$
```

```
                                        ;
                                        ;       NOW FIND THE BN ASSOCIATED WITH THAT COUNT
                                        ;
005102                          5$:     PUSH    <R5,R4>         ; PUSH THE UNIT AND SUBUNIT POINTERS
005102  100465                                                                 MOV R5,-(SP)
005103  100464                                                                 MOV R4,-(SP)
005104  104205  000015                 MOV     #S.TGOF,R5      ; R5 WILL POINT TO THE TRACK/GROUP OFFSETS
005106  105045                         ADD     R4,R5           ; R5 POINTS TO THE TRACK/GROUP ORIGINAL OFFSET
005107  104257                         MOV     (R5)+,R0        ; GET LO ORDER OFFSET, MOVE TO RUNNING TOTAL
005110  104253                         MOV     (R5)+,R3        ; GET HI ORDER OFFSET, MOVE TO RUNNING TOTAL
005111  107201  000001         6$:     SUB     #1,R1           ; DECREMENT LO COUNT BY ONE
005113  045120                         BCC     7$              ; IF NO BORROW, BRANCH
005114  107202  000001                 SUB     #1,R2           ; DECREMENT HI COUNT BY ONE
005116                                 BCS     9$              ; IF COUNT EXHAUSTED, BRANCH
005116  045120                                                                 BCC     .+2
005117  005132                                                                 BR      9$
005120  104254                 7$:     MOV     (R5)+,R4        ; GET T/G OFFSET
005121  055126                         BNE     8$              ; IF NOT END OF LIST, BRANCH
005122  104205  000017                 MOV     #S.TGSS,R5      ; R5 WILL POINT TO START OF T/G LIST
005124  105165                         ADD     (SP),R5         ; R5 POINTS TO START OF T/G LIST
005125  104254                         MOV     (R5)+,R4        ; GET T/G OFFSET
005126  105047                 8$:     ADD     R4,R0           ; ADD TO RUNNING TOTAL
005127  045111                         BCC     6$              ; IF NO CARRY, BRANCH
005130  115403                         INC     R3              ; PROPOGATE CARRY
005131  005111                         BR      6$              ; BRANCH
005132                          9$:     POP     <R4,R5>         ; RESTORE
005132  104264                                                                 MOV (SP)+,R4
005133  104265                                                                 MOV (SP)+,R5
```

```
                                        ;
                                        ;       NOW FIND A RANDOM OFFSET INTO THE RANDOM TRACK OR GROUP
                                        ;
 005134                                 PUSH    R0              ; STORE LO ORDER TOTAL ON STACK
 005134  100467                                                                         MOV R0,-(SP)
 005135  104642  000006                 MOV     S.TRKL(R4),R2   ; R2 HAS TRACK LENGTH
 005137  104147                         MOV     (R4),R0         ; GET SUBUNIT PARAMETERS
 005140                                 ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
 005140  102207  000020                 BIT     #TRACKS,R0      ; SEE IF USING TRACKS
 005142  055156                         BNE     11$             ; IF SO, BRANCH
 005143  104641  000007                 MOV     S.SCHR(R4),R1   ; R1 POINTS TO SUBUNIT PARAMETERS
 005145  104611  000003                 MOV     TRKGRP(R1),R1   ; R1 HAS NUMBER OF TRACKS/GROUP
 005147  103201  177400                 BIC     #HIBYTE,R1      ; CLEAR UNUSED BITS
 005151  114007                         CLR     R0              ; CLEAR RUNNING TOTAL
 005152  105027                 10$:    ADD     R2,R0           ; FIND SECTORS/GROUP
 005153  117401                         DEC     R1              ; DECREMENT COUNT
 005154  055152                         BNE     10$             ; IF UNEXHAUSTED, BRANCH
 005155  104072                         MOV     R0,R2           ; MOVE COUNT TO R2 (MATCH WITH TRACKS ABOVE)
 005156                         11$:    PUSH    R2              ; SAVE ON STACK
 005156  100462                                                                         MOV R2,-(SP)
 005157  117402                         DEC     R2              ; ADJUST COUNT
 005160  104027                         MOV     R2,R0           ; MOVE TO MASKING REGISTER
 005161  025034                         CALL    MASK            ; FIND MASK
 005162  024773                 12$:    CALL    RANDOM          ; GET RANDOM NUMBER
 005163  103071                         BIC     R0,R1           ; STRIP OFF UNUSED BITS
 005164  106021                         CMP     R2,R1           ; SEE IF RANDOM NUMBER SMALL ENOUGH
 005165                                 BCS     12$             ; IF NOT, BRANCH
 005165  045167                                                                         BCC     .+2
 005166  005162                                                                         BR      12$
 005167                                 POP     R2              ; GET NUMBER OF SECTORS
 005167  104262                                                                         MOV (SP)+,R2
 005170  107012                         SUB     R1,R2           ; R2 IS NOW SECTORS REMAINING IN T/G
 005171  105261                                                                         ADD     (SP)+,R1
 005172  100651  000051                 MOV     R1,U.MBN(R5)    ; SAVE
 005174  045176                         BCC     13$             ; IF NO CARRY, BRANCH
 005175  115403                         INC     R3              ; PROPOGATE CARRY
 005176  100653  000052         13$:    MOV     R3,U.MBN+1(R5)  ; SAVE HI ORDER STARTING BN
```

```
                                        ;
                                        ;       NOW COME UP WITH A RANDOM NUMBER OF SECTORS TO R/W WITHOUT GOING
                                        ;       OFF THE TRACK/GROUP THAT WE'RE ON
                                        ;
005200  106202  000016                  CMP     #16,R2          ; ONLY WRITE A MAXIMUM OF 16 SECTORS
005202  035206                          BPL     15$             ; IF LESS THAN 16, BRANCH
005203  104207  177774                  MOV     #MAXMSK,R0      ; SET READ/WRITE LIMIT
005205  005210                          BR      14$             ; BRANCH
005206  104027                  15$:    MOV     R2,R0           ; R0 IS SECTORS LEFT ON THIS T/G
005207  025034                          CALL    MASK            ; GET MASK
005210  024773                  14$:    CALL    RANDOM          ; GET RANDOM NUMBER
005211  103071                          BIC     R0,R1           ; STRIP OFF UNUSED BITS
005212  115401                          INC     R1              ; MUST BE GREATER THAN ZERO
005213  106021                          CMP     R2,R1           ; SEE IF SMALL ENOUGH
005214                                  BCS     14$             ; IF NOT, BRANCH
005214  045216                                                                  BCC     .+2
005215  005210                                                                  BR      14$
005216  100651  000023                  MOV     R1,U.TSEC(R5)   ; SAVE SECTORS TO WRITE
005220  104207  004641                  MOV     #AFTOP,R0       ; NEXT MODULE IS AFTOP
005222  114001                          CLR     R1              ; IMMIDATE CALL
005223  114002                          CLR     R2              ; NO ERRORS
005224  003231                          BR      JMPRET          ; RETURN TO                 21
                                        .IF     LE,MAXADR-.
                                MAXADR  =       .+1
                                        .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE        GET NEXT        AL TRACK/GROUP SECTOR
005225                                  DMOVLY  ST,AREA2
005225  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
                                ;
000006                          RNDTG+1
                                ;
                                ;       FIND THE NEXT TRACK/GROUP TO TEST        ALLY
                                ;
                                .ENABL  LSB
005052  104657  000046          MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
005054  104141                  MOV     (R4),R1         ; GET SUBUNIT PARAMETERS
005055                          ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
005055  102207  010000          BIT     #DIREC,R0       ; SEE IF GOING UP (INITIAL WRITE) OR DOWN
005057  015067                  BEQ     3$              ; IF GOING UP OR INITIAL WRITE, BRANCH
005060  102201  000020          BIT     #TRACKS,R1      ; SEE IF TRACKS OR GROUPS
005062  015065                  BEQ     2$              ; IF GROUPS, BRANCH
005063  025207                  CALL    DOWNT           ; GO DOWN A TRACK
005064  005075                  BR      5$              ; CALCULATE NUMBER OF SECTORS TO READ/WRITE
005065  025351          2$:     CALL    DOWNG           ; GO DOWN A GROUP
005066  005075                  BR      5$              ; CALCULATE NUMBER OF SECTORS TO READ/WRITE
005067  102201  000020  3$:     BIT     #TRACKS,R1      ; SEE IF TRACKS OR GROUPS
005071  015074                  BEQ     4$              ; IF GROUPS, BRANCH
005072  025130                  CALL    UPT             ; GO UP A TRACK
005073  005075                  BR      5$              ; CALCULATE NUMBER OF SECTORS TO READ/WRITE
005074  025271          4$:     CALL    UPG             ; GO UP A GROUP
005075  104642  000006  5$:     MOV     S.TRKL(R4),R2   ; GET SECTORS/TRACK
005077  104141                  MOV     (R4),R1         ; GET SUBUNIT PARAMETERS
005100                          ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
005100  115000  002223          TST     M.PARM          ; SEE IF INITIAL WRITE IN PROGRESS
005102  035121                  BPL     7$              ; IF NOT, BRANCH
005103                          ASSUME  IWIPRG,100000   ; ASSUME IWIPRG IS SIGN BIT
005103  102201  000020          BIT     #TRACKS,R1      ; SEE IF TRACKS IN USE
005105  055121                  BNE     7$              ; IF SO, BRANCH
005106  104027                  MOV     R2,R0           ; COPY SECTORS/TRACK TO R0
005107  104641  000007          MOV     S.SCHR(R4),R1   ; GET POINTER TO SUBUNIT CHARACTERISTICS
005111  104611  000003          MOV     TRKGRP(R1),R1   ; GET TRACKS/GROUP
005113  103201  177400          BIC     #HIBYTE,R1      ; CLEAR UNUSED BITS
005115  114002                  CLR     R2              ; CLEAR RUNNING TOTAL
005116  105072          6$:     ADD     R0,R2           ; ADD TO RUNNING TOTAL
005117  117401                  DEC     R1              ; DECREMENT COUNT
005120  055116                  BNE     6$              ; IF INCOMPLETE, BRANCH
005121  100652  000027  7$:     MOV     R2,U.TSEC(R5)   ; SAVE NUMBER OF SECTORS TO PROCESS
005123  104207  004641  8$:     MOV     #AFTOP,R0       ; NEXT MODULE IS AFTOP
005125  114001                  CLR     R1              ; IMMIDATE CALL
005126  114002                  CLR     R2              ; NO ERRORS
005127  003231                  BR      JMPRET          ; RETURN TO                52
```

```
                                        .SBTTL  UPT     - MOVE UP ONE TRACK
005130                                  UPT:
                                        ;
                                        ;       POINT U.MBN TO THE NEXT TRACK TO TEST (IT DOESN'T HAVE TO BE A
                                        ;       TRACK, BUT IF GROUPS, U.MBN MUST BE ADJUSTED BACK TO THE FIRST
                                        ;       BN IN THE GROUP
                                        ;
005130  115000  002217                  10$:    TST     SCR1            ; SEE IF NEW SUBUNIT
005132  015135                                  BEQ     10$             ; IF NOT, BRANCH
005133  025405                                  CALL    NEWUPT          ; SETUP NEW SUBUNIT
005134  005206                                  BR      4$              ; EXIT
005135  104657  000015                  10$:    MOV     U.CCNT(R5),R0   ; GET LO ORDER COUNT
005137  107207  000001                          SUB     #1,R0           ; DECREMENT
005141  100657  000015                          MOV     R0,U.CCNT(R5)   ; SAVE
005143  045162                                  BCC     2$              ; IF NO BORROW, BRANCH
005144  104657  000016                          MOV     U.CCNT+1(R5),R0 ; GET HI ORDER COUNT
005146  107207  000001                          SUB     #1,R0           ; PROPOGATE BORROW
005150  045160                                  BCC     1$              ; IF NO BORROW, BRANCH
005151  104657  000046                          MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
005153  101207  004000                          BIS     #NEWSUB,R0      ; FLAG AS TO GO ONTO NEXT SUBUNIT
005155  100657  000046                          MOV     R0,U.PARM(R5)   ; SAVE
005157  005206                                  BR      4$              ; EXIT
005160  100657  000016                  1$:     MOV     R0,U.CCNT+1(R5) ; SAVE
005162  104657  000017                  2$:     MOV     U.PCTG(R5),R0   ; GET POINTER TO CURRENT TRACK/GROUP
005164  104271                                  MOV     (R0)+,R1        ; GET OFFSET TO NEXT TRACK
005165  055172                                  BNE     3$              ; IF OFFSET, BRANCH
005166  104207  000017                          MOV     #S.TGSS,R0      ; R0 WILL POINT TO START OF OFFSETS
005170  105047                                  ADD     R4,R0           ; R0 POINTS TO OFFSETS
005171  104271                                  MOV     (R0)+,R1        ; R1 HAS OFFSET
005172  100657  000017                  3$:     MOV     R0,U.PCTG(R5)   ; SAVE POINTER TO CURRENT TRACK/GROUP
005174  105651  000051                          ADD     U.MBN(R5),R1    ; ADD OFFSET TO CURRENT BN
005176  100651  000051                          MOV     R1,U.MBN(R5)    ; SAVE
005200  045206                                  BCC     4$              ; IF NO CARRY, EXIT
005201  104651  000052                          MOV     U.MBN+1(R5),R1  ; GET HI ORDER BN
005203  115401                                  INC     R1              ; PROPOGATE CARRY
005204  100651  000052                          MOV     R1,U.MBN+1(R5)  ; SAVE
005206  000000                  4$:     RETURN                          ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  DOWNT  - MOVE DOWN ONE TRACK
005207                                  DOWNT:
                                        ;
                                        ;       MOVE THE MASTER BN DOWN TO TEST THE PROCEEDING TRACK/GROUP
                                        ;
005207  115000  002217                  TST     SCR1            ; SEE IF NEW SUBUNIT
005211  015214                          BEQ     10$             ; IF NOT, BRANCH
005212  025433                          CALL    NEWDNT          ; INITIALIZE THIS SUBUNIT
005213  005270                          BR      5$              ; EXIT
005214  104657  000015          10$:    MOV     U.CCNT(R5),R0   ; GET LO ORDER COUNT
005216  107207  000001                  SUB     #1,R0           ; DECREMENT COUNT
005220  100657  000015                  MOV     R0,U.CCNT(R5)   ; SAVE
005222  045241                          BCC     2$              ; IF NO BORROW, BRANCH
005223  104657  000016                  MOV     U.CCNT+1(R5),R0 ; GET HI ORDER COUNT
005225  107207  000001                  SUB     #1,R0           ; PROPOGATE BORROW
005227  045237                          BCC     1$              ; IF NO CARRY, BRANCH
005230  104657  000046                  MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
005232  101207  004000                  BIS     #NEWSUB,R0      ; FLAG AS TO GO ONTO NEXT SUBUNIT
005234  100657  000046                  MOV     R0,U.PARM(R5)   ; SAVE
005236  005270                          BR      5$              ; EXIT
005237  100657  000016          1$:     MOV     R0,U.CCNT+1(R5) ; SAVE
005241  104201  000017          2$:     MOV     #S.TGSS,R1      ; R0 WILL POINT TO START OF T/G OFFSETS
005243  105041                          ADD     R4,R1           ; R0 POINTS TO START OF T/G OFFSETS
005244  104657  000017                  MOV     U.PCTG(R5),R0   ; GET POINTER INTO LIST
005246  106071                          CMP     R0,R1           ; SEE IF AT START OF LIST
005247  055253                          BNE     4$              ; IF NOT, BRANCH
005250  115407          3$:             INC     R0              ; R0 POINTS TO NEXT WORD
005251  104171                          MOV     (R0),R1         ; SEE IF END OF LIST
005252  055250                          BNE     3$              ; IF NOT, BRANCH
005253  104651  000051          4$:     MOV     U.MBN(R5),R1    ; GET LO ORDER MASTER BN
005255  107471                          SUB     -(R0),R1        ; SUBTRACT OFFSET
005256  100651  000051                  MOV     R1,U.MBN(R5)    ; SAVE
005260  100657  000017                  MOV     R0,U.PCTG(R5)   ; SAVE POINTER
005262  045270                          BCC     5$              ; IF NO BORROW, EXIT
005263  104657  000052                  MOV     U.MBN+1(R5),R0  ; GET HI ORDER WORD
005265  117407                          DEC     R0              ; PROPOGATE BORROW
005266  100657  000052                  MOV     R0,U.MBN+1(R5)  ; SAVE
005270  000000          5$:             RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  UPG     - MOVE UP ONE GROUP
005271                                  UPG:
                                        ;
                                        ;       MOVE MASTER BN UP TO THE NEXT GROUP
                                        ;
005271  115000  002217                  TST     SCR1            ; SEE IF THIS IS A NEW SUBUNIT
005273  015277                          BEQ     10$             ; IF NOT, BRANCH
005274  025405                          CALL    NEWUPT          ; INITILIZE THE SUBUNIT FOR TRACKS
005275  025507                          CALL    SETSEC          ; INITILIZE THE SUBUNIT FOR GROUPS
005276  005350                          BR      5$              ; EXIT
005277  102201  040000          10$:    BIT     #INITW,R1       ; SEE IF AN INITIAL WRITE IS BEING DONE
005301  055346                          BNE     4$              ; IF SO, BRANCH
005302  104641  000006                  MOV     S.TRKL(R4),R1   ; R1 HAS TRACK LENGTH
005304  104657  000020                  MOV     U.CTRK(R5),R0   ; GET TRACK COUNT
005306  117407                          DEC     R0              ; DECREMENT COUNT
005307  015325                          BEQ     1$              ; IF COUNT EXHAUSTED, BRANCH
005310  100657  000020                  MOV     R0,U.CTRK(R5)   ; SAVE
005312  105651  000051                  ADD     U.MBN(R5),R1    ; MOVE BN TO NEXT TRACK
005314  100651  000051                  MOV     R1,U.MBN(R5)    ; SAVE
005316  045350                          BCC     5$              ; EXIT
005317  104657  000052                  MOV     U.MBN+1(R5),R0  ; GET HI ORDER BN
005321  115407                          INC     R0              ; PROPOGATE CARRY
005322  100657  000052                  MOV     R0,U.MBN+1(R5)  ; SAVE
005324  005350                          BR      5$              ; EXIT
005325  025507                  1$:     CALL    SETSEC          ; SETUP NEXT TRACK/GROUP COUNT
005326  114002                          CLR     R2              ; CLEAR RUNNING TOTAL
005327  117407                  2$:     DEC     R0              ; DECREMENT COUNT
005330  015333                          BEQ     3$              ; IF COUNT EXPIRED, BRANCH
005331  105012                          ADD     R1,R2           ; ADD SECTORS/TRACK TO RUNNING TOTAL
005332  005327                          BR      2$              ; LOOP
005333  104657  000051          3$:     MOV     U.MBN(R5),R0    ; GET LO ORDER MASTER BN
005335  107027                          SUB     R2,R0           ; ADJUST BACK TO START OF GROUP
005336  100657  000051                  MOV     R0,U.MBN(R5)    ; SAVE
005340  045346                          BCC     4$              ; IF NO BORROW, BRANCH
005341  104657  000052                  MOV     U.MBN+1(R5),R0  ; GET HI ORDER MASTER BN
005343  117407                          DEC     R0              ; PROPOGATE BORROW
005344  100657  000052                  MOV     R0,U.MBN+1(R5)  ; SAVE
005346  114007                  4$:     CLR     R0              ; CLEAR R0 SO UPT DOESN'T DETECT A NEW SUBUNIT
005347  025130                          CALL    UPT             ; GO TO NEXT GROUP
005350  000000                  5$:     RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  DOWNG  - MOVE DOWN ONE GROUP
005351                                  DOWNG:
                                        ;
                                        ;       MOVE MASTER BN TO THE NEXT LOWER GROUP
                                        ;
005351  115000  002217                  TST     SCR1            ; SEE IF THIS IS A NEW SUBIUNIT
005353  015357                          BEQ     10$             ; IF NOT, BRANCH
005354  025433                          CALL    NEWDNT          ; SETUP THIS SUBUNIT FOR TRACKS
005355  025520                          CALL    LSTTRK          ; SETUP THIS SUBUNIT FOR GROUPS
005356  005404                          BR      4$              ; EXIT
005357  104657  000020          10$:    MOV     U.CTRK(R5),R0   ; GET TRACK COUNT
005361  117407                          DEC     R0              ; DECREMENT COUNT
005362  015402                          BEQ     1$              ; IF EXPIRED, BRANCH
005363  100657  000020                  MOV     R0,U.CTRK(R5)   ; SAVE
005365  104657  000051                  MOV     U.MBN(R5),R0    ; GET LO ORDER MASTER BN
005367  107647  000006                  SUB     S.TRKL(R4),R0   ; SUBTRACT TRACK LENGTH
005371  100657  000051                  MOV     R0,U.MBN(R5)    ; SAVE
005373  045404                          BCC     4$              ; EXIT
005374  104657  000052                  MOV     U.MBN+1(R5),R0  ; GET HI MASTER BN
005376  117407                          DEC     R0              ; PROPOGATE BORROW
005377  100657  000052                  MOV     R0,U.MBN+1(R5)  ; SAVE
005401  005404                          BR      4$              ; EXIT
005402  025207                  1$:     CALL    DOWNT           ; GO DOWN A GROUP
005403  025520                          CALL    LSTTRK          ; SET MASTER BN ON LAST TRACK OF GROUP
005404  000000                  4$:     RETURN                  ; RETURN TO CALLING PROGRM
```

```
UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE 89
NEWUPT - INITILIZE PARAMETERS FOR         ALLY UP BY TRACKS

                                        .SBTTL  NEWUPT - INITILIZE PARAMETERS FOR         ALLY UP BY TRACKS
005405                                  NEWUPT:
                                        ;
                                        ;       INITIALIZE THIS SUBUNIT FOR         ALLY ACCESSING TRACKS IN
                                        ;       ASCENDING ORDER
                                        ;
005405  104647  000013                          MOV     S.MCNT(R4),R0   ; GET MAXIMUM COUNT
005407  100657  000015                          MOV     R0,U.CCNT(R5)   ; SAVE IN CURRENT COUNT
005411  104647  000014                          MOV     S.MCNT+1(R4),R0 ; GET HI ORDER MAXIMUM COUNT
005413  100657  000016                          MOV     R0,U.CCNT+1(R5) ; SAVE IN HI ORDER CURRENT COUNT
005415  104647  000015                          MOV     S.TGOF(R4),R0   ; GET STARTING OFFSET
005417  100657  000051                          MOV     R0,U.MBN(R5)    ; SAVE IN MASTER BN
005421  104647  000016                          MOV     S.TGOF+1(R4),R0 ; GET HI ORDER STARTING OFFSET
005423  100657  000052                          MOV     R0,U.MBN+1(R5)  ; SAVE IN HI ORDER MASTER BN
005425  104207  000017                          MOV     #S.TGSS,R0      ; R0 WILL POINT TO START OF OFFSETS
005427  105047                                  ADD     R4,R0           ; R0 POINTS TO START OF OFFSETS
005430  100657  000017                          MOV     R0,U.PCTG(R5)   ; SAVE IN POINTER TO CURRENT TRACK/GROUP
005432  000000                                  RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                .SBTTL  NEWDNT - INITILIZE PARAMETERS FOR        ALLY DOWN BY TRACKS
005433                          NEWDNT:
                                ;
                                ;       INITIALIZE THIS SUBUNIT FOR        ALLY ACCESSING TRACKS IN
                                ;       DESCENDING ORDER
                                ;
005433                                  PUSH    <R5,R4>         ; SAVE UNIT AND SUBUNIT POINTERS
005433  100465                                                                  MOV R5,-(SP)
005434  100464                                                                  MOV R4,-(SP)
005435  104642  000015                 MOV     S.TGOF(R4),R2   ; R2 HAS LO ORDER STARTING OFFSET
005437  104643  000016                 MOV     S.TGOF+1(R4),R3 ; R3 HAS HI ORDER STARTING OFFSET
005441  104647  000013                 MOV     S.MCNT(R4),R0   ; R0 HAS LO ORDER MAXIMUM COUNT
005443  100657  000015                 MOV     R0,U.CCNT(R5)   ; SAVE LO ORDER MAXIMUM COUNT
005445  104641  000014                 MOV     S.MCNT+1(R4),R1 ; R1 HAS HI ORDER MAXIMUM COUNT
005447  100651  000016                 MOV     R1,U.CCNT+1(R5) ; SAVE HI ORDER MAXIMUM COUNT
005451  105204  000017                 ADD     #S.TGSS,R4      ; R4 POINTS TO START OF OFFSETS
005453  107207  000001         1$:     SUB     #1,R0           ; DECREMENT COUNT
005455  045462                         BCC     2$              ; IF NO CARRY, BRANCH
005456  107201  000001                 SUB     #1,R1           ; PROPOGATE BORROW
005460                                  BCS     4$              ; IF COUNT EXHAUSTED, BRANCH
005460  045462                                                                  BCC     .+2
005461  005474                                                                  BR      4$
005462  104245                 2$:     MOV     (R4)+,R5        ; GET OFFSET
005463  055470                         BNE     3$              ; IF NOT END-OF-LIST, BRANCH
005464  104204  000017                 MOV     #S.TGSS,R4      ; R4 WILL POINT TO START OF OFFSETS
005466  105164                         ADD     (SP),R4         ; R4 POINTS TO START OF OFFSETS
005467  104245                         MOV     (R4)+,R5        ; GET OFFSET
005470  105052                 3$:     ADD     R5,R2           ; ADD OFFSET TO BN
005471  045453                         BCC     1$              ; IF NO CARRY, LOOP
005472  115403                         INC     R3              ; PROPOGATE CARRY
005473  005453                         BR      1$              ; LOOP
005474  104665  000001         4$:     MOV     1(SP),R5        ; GET UNIT POINTER
005476  100654  000017                 MOV     R4,U.PCTG(R5)   ; SAVE POINTER TO CURRENT TRACK/GROUP
005500                                  POP     <R4,R5>         ; RESTORE
005500  104264                                                                  MOV (SP)+,R4
005501  104265                                                                  MOV (SP)+,R5
005502  100652  000051                 MOV     R2,U.MBN(R5)    ; SAVE LO ORDER STARTING BN
005504  100653  000052                 MOV     R3,U.MBN+1(R5)  ; SAVE HI ORDER
005506  000000                         RETURN                  ; RETURN TO CALLING PROGRAM
```

```
      1                                          .SBTTL  SETSEC - SETUP TRACK/GROUP COUNT FOR SELECTED GROUPS
      2 005507                                   SETSEC:
      3                                          ;
      4                                          ;       MOVE TRACKS/GROUP TO U.CTRK
      5                                          ;
      6 005507  104647  000007                           MOV     S.SCHR(R4),R0   ; R0 POINTS TO SUBUNIT CHARACTERISTICS
      7 005511  104677  000003                           MOV     TRKGRP(R0),R0   ; R0 HAS TRACKS/GROUPS
      8 005513  103207  177400                           BIC     #HIBYTE,R0      ; CLEAR UNUSED BITS
      9 005515  100657  000020                           MOV     R0,U.CTRK(R5)   ; SAVE AS TRACK COUNT
     10 005517  000000                                   RETURN                  ; RETURN TO CALLING MODULE
```

LSTTRK - SET MASTER BN TO POINT TO LAST TRACK IN THE GROUP

```
                                        .SBTTL  LSTTRK - SET MASTER BN TO POINT TO LAST TRACK IN THE GROUP
005520                                  LSTTRK:
                                        ;
                                        ;       SET MBN TO LAST TRACK OF GROUP
                                        ;
005520  025507                                  CALL    SETSEC          ; SETUP TRACK/GROUP COUNT
005521  114002                                  CLR     R2              ; CLEAR RUNNING TOTAL
005522  117407                          2$:     DEC     R0              ; DECREMENT COUNT
005523  015527                                  BEQ     3$              ; IF COUNT EXPIRED, BRANCH
005524  105642  000006                          ADD     S.TRKL(R4),R2   ; ADD SECTORS/TRACK TO RUNNING LENGTH
005526  005522                                  BR      2$              ; LOOP
005527  105652  000051                  3$:     ADD     U.MBN(R5),R2    ; MBN WILL POINT TO LAST TRACK IN GROUP
005531  100652  000051                          MOV     R2,U.MBN(R5)    ; SAVE
005533  045537                                  BCC     4$              ; IF NO CARRY, EXIT
005534  104657  000052                          MOV     U.MBN+1(R5),R0  ; GET HI ORDER BN
005536  115407                                  INC     R0              ; PROPOGATE CARRY
005537  100657  000052                  4$:     MOV     R0,U.MBN+1(R5)  ; SAVE
005541  000000                                  RETURN                  ; RETURN TO CALLING MODULE
                                                .IF     LE,MAXADR-.
                                        MAXADR  =       .+1
                                                .ENDC
```

```
                                        .SBTTL  ********** OVERLAY MODULE BUILDP - BUILD THE READ/WRITE LINKED LISTS
 005542                                         DMOVLY  BP,AREA0
 005542  000105                                .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;
                                        ;
 000007                                 BUILDP  =                       ; READ/WRITE LINKED LIST BUILDER
                                        ;
                                        ;       BUILD THE READ/WRITE LINKED LISTS
                                        ;
                                                .ENABL  LSB
 004422  021031                                 CALL    TLKHST          ; COMMUNICATE WITH THE HOST SO NOT TO TIME OUT
 004423  104657  000046                        MOV     U.PARM(R5),R0   ; R0 HAS UNIT PARAMETERS
 004425  104650  000022  002235                MOV     U.MSEC(R5),MAXSEC       ; GET NUMBER OF SECTORS TO READ/WRITE
 004430  102207  000200                        BIT     #RBNBN,R0       ; SEE IF RBN TO BE READ/WRITTEN
 004432  014442                                BEQ     1$              ; IF NOT, BRANCH
 004433  104650  000055  002175                MOV     U.RBN(R5),CURBN ; MOVE LO ORDER TO CURRENT BN
 004436  104650  000056  002176                MOV     U.RBN+1(R5),CURBN+1     ; MOVE HI ORDER TO CURRENT BN
 004441  004450                                BR      4$              ; BRANCH
 004442  104650  000053  002175  1$:           MOV     U.CBN(R5),CURBN         ; MOVE LO ORDER TO CURRENT BN
 004445  104650  000054  002176                MOV     U.CBN+1(R5),CURBN+1     ; MOVE LO ORDER TO CURRENT BN
 004450  104300  002227  004710  4$:           MOV     MEMPOL,TMEMPL   ; START OF BUILD (INITILIZE MEM POOL)
 004453  114007                                CLR     R0              ; FORCE THE CALCULATION TABLE TO BE FILLED
 004454  104070  002232                        MOV     R0,CHAINS       ; MARK CHAIN AS EMPTY
 004456  022027                        5$:     CALL    CALC            ; CALCULATE CYL, TRK AND GRP
 004457  106650  000064  002201                CMP     U.CCYL(R5),CYL  ; SEE IF LO CYL MATCHES
 004462  054663                                BNE     7$              ; IF NOT, BRANCH
 004463  106650  000065  002202                CMP     U.CCYL+1(R5),CYL+1      ; SEE IF HI CYL MATCHES
 004466  054663                                BNE     7$              ; IF NOT, BRANCH
 004467  106650  000066  002203                CMP     U.CGRP(R5),GROUP        ; SEE IF GROUP MATCHES
 004472  054663                                BNE     7$              ; IF NOT, BRANCH
```

```
                                        .SBTTL  LINK   - BUILD A LINK (NODE) IN THE READ/WRITE CHAIN
                                        ;LINK
                                        ;
                                        ;       LINK WILL CREATE THE LINKS FOR THE CHAIN
                                        ;
 004473  104653  000046                        MOV     U.PARM(R5),R3   ; R3 HAS UNIT PARAMETERS
 004475  104307  002232                        MOV     CHAINS,R0       ; R0 POINTS TO START OF CHAIN
 004477  054521                                BNE     22$             ; IF THE CHAIN IS ALLREADY STARTED, BRANCH
 004500  102203  000100                        BIT     #REDWRT,R3      ; SEE IF READ IN PROGRESS
 004502  014513                                BEQ     21$             ; IF SO, BRANCH
 004503  102203  000400                        BIT     #REVEC,R3       ; SEE IF TRYING TO FIND A REPLACEMENT
 004505  054513                                BNE     21$             ; IF SO, YOU'RE TRYING TO READ, SO BRANCH
 004506  104207  000401                        MOV     #WBUFLN,R0      ; R0 HAS NUMBER OF WORDS IN WRITE BUFFER
 004510  024703                                CALL    ALLOCM          ; ALLOCATE THE MEMORY
 004511  104070  004711                        MOV     R0,SECPTR       ; SECPTR POINTS TO WRITE BUFFER
 004513  104207  000007          21$:          MOV     #LINKLN,R0      ; R0 HAS LENGTH OF CHAIN LINK
 004515  024703                                CALL    ALLOCM          ; ALLOCATE THE MEMORY
 004516  104070  002232                        MOV     R0,CHAINS       ; CHAINS POINTS TO FIRST LINK
 004520  004541                                BR      24$             ; BRANCH
 004521  104171                  22$:          MOV     (R0),R1         ; R1 HAS NEXT LINK POINTER
 004522  102201  100000                        BIT     #EOC,R1         ; SEE IF THIS LINK IS THE LAST
 004524  054531                                BNE     23$             ; IF SO, BRANCH
 004525  103201  170000                        BIC     #^CHBHINB,R1    ; CLEAR UNUSED BITS
 004527  104017                                MOV     R1,R0           ; R0 POINTS TO NEXT LINK
 004530  004521                                BR      22$             ; LOOP
 004531  104072                  23$:          MOV     R0,R2           ; SAVE R0
 004532  104207  000007                        MOV     #LINKLN,R0      ; R0 HAS LENGTH OF LINK
 004534  024703                                CALL    ALLOCM          ; ALLOCATE MEMORY FOR LINK
 004535  103201  100000                        BIC     #EOC,R1         ; CLEAR THE END-OF-CHAIN FLAG
 004537  101071                                BIS     R0,R1           ; PUT IN POINTER TO NEXT LINK
 004540  100121                                MOV     R1,(R2)         ; PUT POINTER BACK IN LAST LINK
 004541                           24$:
```

```
                                        .SBTTL  FILLIN - FILL THE READ/WRITE CHAIN LINK (NODE) WITH REQUIRED INFORMATION
                                        ;FILLIN
                                        ;
                                        ;       FILLIN BUILDS THE PARAMETERS THE UDA REQUIRES TO WRITE OR READ A BLOCK
                                        ;
004541  104302  002204                  MOV     TRACK,R2        ; GET TRACK (HEAD) NUMBER
004543  102203  000100                  BIT     #REDWRT,R3      ; SEE IF READ IN PROGRESS
004545  014562                          BEQ     11$             ; IF SO, BRANCH
004546  102203  000400                  BIT     #REVEC,R3       ; SEE IF TRYING TO FIND A REPLACEMENT
004550  054562                          BNE     11$             ; IF SO, YOU'RE TRYING TO READ, SO BRANCH
004551  101202  122400                  BIS     #WREAL,R2       ; BUILD WRITE REAL TIME COMMAND
004553  100672  000004                  MOV     R2,RW.CMD(R0)   ; MOVE TO SDI REAL TIME COMMAND AREA
004555  104202  140000                  MOV     #WSTOP,R2       ; MOVE LAST BLOCK FLAG TO R2
004557  104301  004711                  MOV     SECPTR,R1       ; R1 POINTS TO WRITE BUFFER
004561  004576                          BR      12$             ; BRANCH
004562  101202  013400          11$:    BIS     #RREAL,R2       ; BUILD READ REAL TIME COMMAND
004564  100672  000004                  MOV     R2,RW.CMD(R0)   ; MOVE TO SDI REAL TIME COMMAND AREA
004566  104202  100000                  MOV     #RSTOP,R2       ; MOVE LAST BLOCK FLAG TO R2
004570                                  PUSH    R0              ; SAVE R0
004570  100467                                                                  MOV R0,-(SP)
004571  104207  000415                  MOV     #RBUFLN,R0      ; SIZE OF READ BUFFER
004573  024703                          CALL    ALLOCM          ; ALLOCATE BUFFER
004574  104071                          MOV     R0,R1           ; R1 POINTS TO BUFFER
004575                                  POP     R0              ; RESTORE R0
004575  104267                                                                  MOV (SP)+,R0
004576  100172                  12$:    MOV     R2,(R0)         ; MOVE LAST BLOCK FLAG TO NEXT BLOCK POINTER
004577                                  ASSUME  RW.STAT,0
004577  100671  000001                  MOV     R1,RW.BUF(R0)   ; MOVE POINTER TO SECTOR TO POINTER AREA
004601  104202  002207                  MOV     #DUMSDI,R2      ; R2 POINTS TO DUMMY SDI AREA
004603  100672  000005                  MOV     R2,RW.SDI(R0)   ; MOVE R2 TO POINTER TO DUMMY SDI AREA
004605  104302  002205                  MOV     SECTOR,R2       ; R2 CONTAINS SECTOR TO BE WRITTEN
004607  100672  000006                  MOV     R2,RW.ANG(R0)   ; MOVE TO ANGLE FROM INDEX
004611  104302  002175                  MOV     CURBN,R2        ; MOVE LOW ORDER BN TO R2
004613  100672  000002                  MOV     R2,RW.LOW(R0)   ; MOVE LOW ORDER BN TO LOW EXPECTED HEADER
004615  104302  002176                  MOV     CURBN+1,R2      ; MOVE HIGH ORDER BN TO R2
004617  102203  000200                  BIT     #RBNBN,R3       ; SEE IF BN IS AN RBN
004621  054631                          BNE     15$             ; IF SO, BRANCH
004622  104143                          MOV     (R4),R3         ; GET SUBUNIT PARAMETERS
004623                                  ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
004623  102203  020000                  BIT     #DCYLS,R3       ; SEE IF USING DIAGNOSTIC CYLINDERS
004625  014633                          BEQ     16$             ; IF NOT, BRANCH
004626  101202  140000                  BIS     #HD.DBN,R2      ; DIAGNOSTIC HEADER
004630  004633                          BR      16$             ; BRANCH
004631  101202  060000          15$:    BIS     #HD.RBN,R2      ; REVECTORED HEADER
004633  105302  002172          16$:    ADD     HIBN,R2         ; ADD HI BN BITS
004635                                  ASSUME  HD.LBN,0
004635  100672  000003                  MOV     R2,RW.HI(R0)    ; MOVE R2 TO HI WORD EXPECTED HEADER
004637  104302  002206                  MOV     INDEX,R2        ; GET DISTANCE FROM INDEX
004641  115402                          INC     R2              ; ADJUST INDEX
004642  106302  002217                  CMP     SCR1,R2         ; SEE IF FIRST SECTOR AFTER INDEX
004644  054646                          BNE     13$             ; IF NOT, BRANCH
004645  114002                          CLR     R2              ; ZERO THETA FROM INDEX
004646  100672  000006          13$:    MOV     R2,RW.ANG(R0)   ; SAVE
004650  107200  000001  002235          SUB     #1,MAXSEC       ; DECREMENT SECTOR COUNT
004653  014663                          BEQ     7$              ; IF COUNT COMPLETE, BRANCH
004654  105200  000001  002175          ADD     #1,CURBN        ; ADD ONE TO LO CURRENT SECTOR
004657  044456                          BCC     5$              ; IF NO CARRY, BRANCH
```

```
     56 004660  115400  002176                    INC     CURBN+1          ; PROPOGATE CARRY
     57 004662  004456                            BR      5$               ; BRANCH
     58 004663  114001                  7$:       CLR     R1               ; IMMIDATE CALL
     59 004664  114002                            CLR     R2               ; NO ERRORS
     60 004665  104653  000046                    MOV     U.PARM(R5),R3    ; GET UNIT PARAMETERS
     61 004667  102203  000100                    BIT     #REDWRT,R3       ; SEE IF READ IS TO BE DONE
     62 004671  014700                            BEQ     2$               ; IF SO, BRANCH
     63 004672  102203  000400                    BIT     #REVEC,R3        ; SEE IF READING RCT
     64 004674  054700                            BNE     2$               ; IF SO, BRANCH
     65 004675  104207  000010                    MOV     #WRITE,R0        ; WRITE ROUTINE IS NEXT
     66 004677  004702                            BR      3$               ; BRANCH
     67 004700  104207  000012          2$:       MOV     #READ,R0         ; READ ROUTINE IS NEXT
     68 004702  003231                  3$:       BR      JMPRET
     69                                           .DSABL  LSB
```

```
                                        .SBTTL  ALLOCM - ALLOCATE MEMORY FOR THE READ/WRITE BUFFERS AND CHAIN
004703                                  ALLOCM:
                                        ;
                                        ;       ALLOCATE MEMORY FOR READ/WRITE BUFFERS
                                        ;
004703  107070  004710                  SUB     R0,TMEMPL       ; SUBTRACT LENGTH FROM MEMORY POOL
004705  104307  004710                  MOV     TMEMPL,R0       ; R0 POINTS TO MEMORY ALLOCATED
004707  000000                          RETURN                  ; RETURN TO CALLING MODULE

004710                                  TMEMPL: .BLKW   1               ; TEMPORARY MEMORY POOL FOR READ/WRITE
004711                                  SECPTR: .BLKW   1               ; SECTOR POINTER FOR WRITE
                                        .IF     LT,BUFARA-.
                                        BUFARA  =       .
                                        .ENDC
                                        .IF     LE,MAXADR-.
                                        MAXADR  =       .+1
                                        .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE WRITE
004712                                  DMOVLY  W,AREA0
004712  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
000010                          WRITE   =       BUILDP+1
                                ;
                                ;       WRITE GENERATES THE PATTERNS AND WRITES THEM TO THE DRIVE
                                ;
                                        .ENABL  LSB
004422  024212                          CALL    DSABLE          ; DISABLE ERROR RECOVERY
004423  104657  000012                  MOV     U.RWTO(R5),R0   ; MOVE READ/WRITE TIMEOUT VALUE TO R0
004425  115407                          INC     R0              ; INCREMENT COUNT
004426  100657  000012                  MOV     R0,U.RWTO(R5)   ; SAVE
004430  014522                          BEQ     8$              ; IF ZERO, WE KNOW THAT THE TIMEOUT IS UNEX
004431  106207  000002                  CMP     #2,R0           ; SEE IF TRIED MAXIMUM TIMES
004433                                  BCS     11$             ; IF SO, BRANCH
004433  044435                                                                  BCC     .+2
004434  004441                                                                  BR      11$
004435  104141                          MOV     (R4),R1         ; GET SUBUNIT PARAMETERS
004436                                  ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
004436  102201  001000                  BIT     #RTRIES,R1      ; SEE IF RETRIES ARE ENABLED
004440  054522                          BNE     8$              ; IF SO, BRANCH
004441                          11$:    HARDER  25,<S.LETR(R4),U.CBN(R5),U.CBN+1(R5)>
004441  104200  002342  001107                                                  MOV     #ER25,HRQ.04
004444  104640  000005  001110                                                  MOV     S.LETR(R4),HRQ.05
004447  104650  000053  001111                                                  MOV     U.CBN(R5),HRQ.06
004452  104650  000054  001112                                                  MOV     U.CBN+1(R5),HRQ.07
004455  104202  107671                                                          MOV     #25!ERHARD+4000.,R2
004457  104020  001105                                                          MOV     R2,HRQ.02
004461  104200  004461  001104                                                  MOV     #.,HRQ.01
004464  104200  060014  001103                                                  MOV     #ERRMC,HRQ.RQ
004467                                  ERRORC  <U.PARM(R5),#RBNTXT,U.RBN(R5),U.RBN+1(R5)>
004467  104650  000046  001113                                                  MOV     U.PARM(R5),HRQ.08
004472  104200  005475  001114                                                  MOV     #RBNTXT,HRQ.09
004475  104650  000055  001115                                                  MOV     U.RBN(R5),HRQ.10
004500  104650  000056  001116                                                  MOV     U.RBN+1(R5),HRQ.11
004503                                  ENDERR  0
004503  114000  002230                                                          CLR     ERRPOS          ; CLEAR THE POSITION
004505  104653  000046                  MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
004507  103203  000200                  BIC     #RBNBN,R3       ; IF HANDLING AN RBN, CLEAR IT
004511  100653  000046                  MOV     R3,U.PARM(R5)   ; SAVE
004513  104201  000001                  MOV     #1,R1           ; ONLY 1 SECTOR HANDLED
004515  100651  000021                  MOV     R1,U.NSEC(R5)   ; STORE
004517  104207  000001                  MOV     #SETUP,R0       ; SETUP IS NEXT MODULE CALLED
004521  004645                          BR      7$              ; BRANCH
004522  021777                  8$:     CALL    BULDUM          ; BUILD DUMMY SDI CONTROL BLOCK

                                .SBTTL  BULDSC - BUILD THE SECTOR TO WRITE (FILL WITH PATTERN AND CALC EDC)
                                ;BULDSC
                                ;
                                ;       BULDSC BUILDS THE INFORMATION AND DATA PATTERN TO BE WRITTEN ON THE
```

```
                                ;       SECTOR
                                ;
004523  104651  000014                MOV     U.PAT(R5),R1    ; GET PATTERN NUMBER TO WRITE
004525  104013                    MOV     R1,R3           ; COPY PATTERN NUMBER TO R3
004526  110701                    SWAB    R1              ; SWAP THE BYTES
004527  101013                    BIS     R1,R3           ; REPLICATE THE PATTERN NUMBER
004530  104031                    MOV     R3,R1           ; COPY TO R1
004531  110201                    ROL     R1              ; ROTATE TO NEXT POSITION
004532  110201                    ROL     R1
004533  110201                    ROL     R1
004534  110201                    ROL     R1
004535  103201  007417            BIC     #^.HBLONB!^CLBLONB,R1   ; CLEAR UNUSED BITS
004537  101013                    BIS     R1,R3           ; REPLICATE THE PATTERN
004540  104307  002232            MOV     CHAINS,R0       ; GET POINTER TO FIRST WRITE CHAIN NODE
004542  104677  000001            MOV     RW.BUF(R0),R0   ; R0 POINTS TO BUFFER AREA
004544  100273                    MOV     R3,(R0)+        ; MOVE PATTERN NUMBERS TO SECTOR
                                .SBTTL  COPPAT - COPY THE DATA PATTERN TO BUFFER
                                ;COPPAT
                                ;
                                ;       COPPAT COPIES THE PATTERN TO THE SECTOR TO BE WRITTEN
                                ;
004545                          PUSH    <R0,R4,R5>      ; SAVE R0,R4,R5
004545  100467                                                          MOV R0,-(SP)
004546  100464                                                          MOV R4,-(SP)
004547  100465                                                          MOV R5,-(SP)
004550  104652  000014                MOV     U.PAT(R5),R2    ; R2 HAS PATTERN NUMBER
004552  104622  002236                MOV     PATPTR(R2),R2   ; POINT TO PATTERN
004554  115402                    INC     R2              ; SKIP EDC
004555  104201  000377            MOV     #SCTWRD,R1      ; R1 HAS NUMBER OF WORDS TO FILL WITH PATTERN
004557  104024            COPLP0: MOV     R2,R4           ; R4 POINTS TO LENGTH OF PATTERN
004560  104243                    MOV     (R4)+,R3        ; R3 CONTAINS LENGTH OF PATTERN
004561  106203  000001            CMP     #1,R3           ; SEE IF PATTERN IS 1 WORD LONG
004563  014573                    BEQ     ONEPAT          ; IF SO, BRANCH
004564  104245            COPLP1: MOV     (R4)+,R5        ; R5 GETS 1 WORD OF THE DATA PATTERN
004565  100275                    MOV     R5,(R0)+        ; MOVE PATTERN WORD TO SECTOR AREA
004566  117401                    DEC     R1              ; DECREMENT NUMBER OF WORDS TO WRITE IN SECTOR
004567  014577                    BEQ     COPFIN          ; IF ALL WORDS WRITTEN, BRANCH
004570  117403                    DEC     R3              ; DECREMENT COUNT OF WORDS IN PATTERN
004571  054564                    BNE     COPLP1          ; IF DATA PATTERN UNFINISHED, BRANCH
004572  004557                    BR      COPLP0          ; BRANCH
004573  104145            ONEPAT: MOV     (R4),R5         ; GET 1 WORD OF DATA PATTERN
004574  100275            COPLP2: MOV     R5,(R0)+        ; MOVE PATTERN TO SECTOR AREA
004575  117401                    DEC     R1              ; DECREMENT NUMBER OF WORDS TO MOVE TO SECTOR
004576  054574                    BNE     COPLP2          ; IF INCOMPLETE, BRANCH
004577  104425            COPFIN: MOV     -(R2),R5        ; GET EDC
004600  100175                    MOV     R5,(R0)         ; MOVE TO BUFFER
004601                          POP     <R5,R4,R0>      ; RESTORE R5,R4,R0
004601  104265                                                          MOV (SP)+,R5
004602  104264                                                          MOV (SP)+,R4
004603  104267                                                          MOV (SP)+,R0
004604  104651  000014                MOV     U.PAT(R5),R1    ; GET PATTERN NUMBER
004606  054613                    BNE     21$             ; IF NOT PATTERN 16, BRANCH
004607  117407                    DEC     R0              ; POINT TO START OF BUFFER
004610  021146                    CALL    CMPEDC          ; COMPUTE EDC
004611  100672  000400            MOV     R2,BF.EDC(R0)   ; SAVE EDC
004613  024646            21$:    CALL    WBLOCK          ; WRITE THE SECTOR
004614  115002                    TST     R2              ; SEE IF ERROR OCCURRED
```

```
    100 004615  054645                          BNE     7$                  ; IF NOT, BRANCH
    101 004616  104643  000011                  MOV     S.MEGW(R4),R3       ; GET MEGABYTE COUNT
    102 004620  106203  003642                  CMP     #1954.,R3           ; SEE IF ONE MEGABYTE TRANSFERED
    103 004622  034641                          BPL     6$                  ; BRANCH IF NOT
    104 004623  107203  003642                  SUB     #1954.,R3           ; CLEAR MEGABYTE COUNT
    105 004625  100643  000011                  MOV     R3,S.MEGW(R4)       ; SAVE NEW COUNT
    106 004627  114000  001105                  CLR     HRQ.02              ; NOT REPORTING READS
    107 004631  104200  000001  001106          MOV     #1,HRQ.03           ; REPORT 1 MEGABIT WRITTEN
    108 004634  104202  060011                  MOV     #T4MXFR,R2          ; SET UP FOR MEGABIT REPORT
    109 004636  104020  001103                  MOV     R2,HRQ.RQ           ; FLAG AS NON-ERROR
    110 004640  004642                          BR      1$                  ; EXIT
    111 004641  114002                  6$:     CLR     R2                  ; NO ERRORS
    112 004642  104207  000011          1$:     MOV     #AFTWRT,R0          ; AFTWRT IS NEXT MODULE
    113 004644  114001                  2$:     CLR     R1                  ; IMMIDATE CALL
    114 004645  003231                  7$:     BR      JMPRET              ; RETURN TO RDWRT
    115                                         .DSABL  LSB
```

```
                                .SBTTL  WBLOCK - WRITE THE SECTOR(S)
004646                          WBLOCK:
                                ;
                                ;       WBLOCK WRITES THE SECTOR(S) ONTO THE DRIVE
                                ;
004646                                  PUSH    R4                      ; SAVE R4
004646  100464                                                                  MOV R4,-(SP)
004647  104307  002232                  MOV     CHAINS,R0               ; POINT TO START OF CHAIN
004651  104652  000025                  MOV     U.MASK(R5),R2           ; R2 HAS SDI INTERCONNECT
004653  104644  000007                  MOV     S.SCHR(R4),R4           ; R4 POINTS TO SUBUNIT CHARACTERISTICS
004655  104644  000005                  MOV     DATPRE(R4),R4           ; R4 CONTAINS DATA PREAMBLE LENGTH
004657  103204  177400                  BIC     #HIBYTE,R4              ; STRIP OFF UNUSED BITS
004661  060012                          XFC     WAITSI                  ; WAIT FOR SECTOR OR INDEX PULSE
004662  115001                          TST     R1                      ; SEE IF ERROR OCCURRED
004663  014723                          BEQ     3$
004664                                  DEVFTL  42,<U.CGRP(R5),U.CCYL(R5),U.CCYL+1(R5)>
004664  104200  003071  001107                                                  MOV     #ER42,HRQ.04
004667  104650  000066  001110                                                  MOV     U.CGRP(R5),HRQ.05
004672  104650  000064  001111                                                  MOV     U.CCYL(R5),HRQ.06
004675  104650  000065  001112                                                  MOV     U.CCYL+1(R5),HRQ.07
004700  104202  047712                                                          MOV     #42!FTLDEV+4000.,R2
004702  104020  001105                                                          MOV     R2,HRQ.02
004704  104200  004704  001104                                                  MOV     #.,HRQ.01
004707  104200  060014  001103                                                  MOV     #ERRMC,HRQ.RQ
004712                                  ENDERR
004712  104200  000051  001113                                                  MOV     #SER22,HRQ.08
004715  104200  000011  002230                                                  MOV     #8+1,ERRPOS     ; SET THE POSITION
004720  024227                          CALL    GOSEEK
004721                                  POP     R4                      ; RESTORE SUBUNIT POINTER
004721  104264                                                                  MOV (SP)+,R4
004722  005024                          BR      6$                      ; EXIT
004723  060003                  3$:     XFC     XWRITE                  ; WRITE THE SECTOR(S)
004724                                  POP     R4                      ; RESTORE R4
004724  104264                                                                  MOV (SP)+,R4
004725  100651  000061                  MOV     R1,U.RWER(R5)           ; SAVE ERROR IF ANY
004727  014776                          BEQ     5$                      ; IF NO ERROR, BRANCH
004730  104651  000026                  MOV     U.WRIT(R5),R1           ; GET WRITE PROTECTION STATUS
                                ;       NOW I OR IN THE READ-ONLY WRITE PROTECT BITS.  THIS WILL CATCH
                                ;       IT IF SOMEHOW I GET INTO THIS MODULE ON A READ-ONLY DRIVE
                                ;       (I HOPE NOT)
004732  101651  000045                  BIS     U.WPRT(R5),R1           ; SET READ-ONLY BITS
004734  104652  000050                  MOV     U.SUBU(R5),R2           ; GET SUBUNIT IN USE
004736  105202  000003                  ADD     #3,R2                   ; ADD 3 TO SHIFT WRITE PROT STAT TO LO NIBBLE
004740  110601                  10$:    ROR     R1                      ; ROTATE PROTECTION STATUS (LO BIT TO CARRY)
004741  117402                          DEC     R2                      ; DECREMENT SUBUNIT IN USE
004742  034740                          BPL     10$                     ; IF >= 0, BRANCH
004743  110601                          ROR     R1                      ; MOVE THE BIT INTO CARRY TO TEST
004744  045023                          BCC     7$                      ; IF CARRY CLEAR (NOT WRITE PROTECTED) BRANCH
004745                                  DEVFTL  18,U.RWER(R5)           ; REPORT WRITE ON WRITE PROTECTED DRIVE
004745  104200  001705  001107                                                  MOV     #ER18,HRQ.04
004750  104650  000061  001110                                                  MOV     U.RWER(R5),HRQ.05
004753  104202  047662                                                          MOV     #18!FTLDEV+4000.,R2
004755  104020  001105                                                          MOV     R2,HRQ.02
004757  104200  004757  001104                                                  MOV     #.,HRQ.01
004762  104200  060014  001103                                                  MOV     #ERRMC,HRQ.RQ
004765                                  ENDERR
004765  104200  000051  001111                                                  MOV     #SER22,HRQ.06
```

```
                004770  104200  000007  002230                                          MOV     #6+1,ERRPOS     ; SET THE POSITION
     38         004773  024227                          CALL    GOSEEK
     39         004774  104051                          MOV     R5,R1           ; DEFFERED CALL
     40         004775  005024                          BR      6$              ; BRANCH
     41         004776  104657  000012          5$:     MOV     U.RWTO(R5),R0   ; GET TIMEOUTS
     42         005000  015023                          BEQ     7$              ; IF NO RETRIES, BRANCH
     43         005001                          REPSFT  SOFT            ; REPORT ONE SOFT ERROR
                005001  104200  000001  001105                                          MOV     #1,HRQ.02
                005004  114000  001106                                                  CLR     HRQ.03
                005006  114000  001107                                                  CLR     HRQ.04
                005010  100467                                                                  MOV R0,-(SP)
                005011  104657  000063                                                  MOV     U.UNUM(R5),R0
                005013  105657  000050                                                  ADD     U.SUBU(R5),R0
                005015  104070  001104                                                  MOV     R0,HRQ.01
                005017  104207  060007                                                  MOV     #T4SOFT,R0
                005021  021053                                                          CALL    HOSTRQ
                005022  104267                                                                  MOV (SP)+,R0
     44         005023  114002          7$:     CLR     R2              ; NO ERRORS
     45         005024  000000          6$:     RETURN                  ; RETURN TO CALLING PROGRAM;
     46                                          .IF     LT,BUFARA-.
     47                                  BUFARA  =       .
     48                                          .ENDC
     49                                          .IF     LE,MAXADR-.
     50                                  MAXADR  =       .+1
     51                                          .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE AFTWRT
                                        DMOVLY  AW,AREA0
005025  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
000011                          AFTWRT  =       WRITE+1
                                ;
                                ;       AFTWRT WILL HANDLE ANY ERRORS THAT OCCURRED DURING THE WRITE
                                ;       OPERATION
                                ;
                                        .ENABL  LSB
                                .SBTTL  FNDWER - IF ERROR DURING WRITE, FIND IT'S POSITION IN THE CHAIN
                                ;FNDWER
                                ;
                                ;       FIND THE FIRST BUFFER THAT IS NOT WRITTEN
                                ;
004422                          PUSH    <R4,R5>         ; SAVE REGISTERS
004422  100464                                                          MOV R4,-(SP)
004423  100465                                                          MOV R5,-(SP)
004424  104654  000053                 MOV     U.CBN(R5),R4    ; GET LO ORDER BN
004426  104655  000054                 MOV     U.CBN+1(R5),R5  ; GET HI BN
004430  114002                         CLR     R2              ; CLEAR SECTOR COUNT
004431  104307  002232                 MOV     CHAINS,R0       ; R0 POINTS TO FIRST BUFFER
004433  100674  000002         31$:    MOV     R4,RW.LOW(R0)   ; MOVE TO CHAIN
004435  100675  000003                 MOV     R5,RW.HI(R0)    ; MOVE TO CHAIN
004437  104171                         MOV     (R0),R1         ; GET STATUS BITS
004440                                 ASSUME  RW.STAT,0
004440  102201  040000                 BIT     #BUFFLG,R1      ; SEE IF THIS BUFFER HAS BEEN WRITTEN
004442  054456                         BNE     32$             ; IF NOT, BRANCH
004443  115402                         INC     R2              ; INCREMENT SECTOR COUNT
004444  115001                         TST     R1              ; SEE IF END-OF-LIST
004445  074456                         BMI     32$             ; IF LAST BUFFER, EXIT
004446                                 ASSUME  WSTOP-40000,100000      ; ASSUME WRITE STOP BIT IS SIGN BIT
004446  103201  170000                 BIC     #^CHBHINB,R1    ; CLEAR UNUSED BITS
004450  104017                         MOV     R1,R0           ; MOVE TO R0
004451  105204  000001                 ADD     #1,R4           ; ADD ONE TO LOW ORDER BN
004453  044433                         BCC     31$             ; IF NO CARRY, BRANCH
004454  115405                         INC     R5              ; PRPOGATE CARRY
004455  004433                         BR      31$             ; LOOP
004456                          32$:    POP     <R5,R4>         ; RESTORE REGISTERS
004456  104265                                                          MOV (SP)+,R5
004457  104264                                                          MOV (SP)+,R4
004460  100652  000021                 MOV     R2,U.NSEC(R5)   ; SAVE AS NUMBER OF SECTORS HANDLED
004462  105642  000011                 ADD     S.MEGW(R4),R2   ; ADD TO MEGABITS WRITTEN
004464  100642  000011                 MOV     R2,S.MEGW(R4)   ; SAVE
004466  104652  000061                 MOV     U.RWER(R5),R2   ; SEE IF ANY ERROR OCCURRED
004470  054503                         BNE     3$              ; IF SO, BRANCH
004471  104657  000046                 MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
004473  103207  000200                 BIC     #RBNBN,R3       ; CLEAR RBN BIT
004475  100657  000046                 MOV     R0,U.PARM(R5)   ; SAVE
004477  104207  000001                 MOV     #SETUP,R0       ; SETUP NEXT MODULE
004501  104051                         MOV     R5,R1           ; DEFERRED CALL
```

```
     60 004502  005223                        BR      17$             ; EXIT
     61 004503  021725              3$:       CALL    BLKCHK          ; SEE IF THIS IS A KNOWN BAD BLOCK
     62 004504                                BCS     15$             ; IF NOT, BRANCH
        004504  044506                                                                  BCC     .+2
        004505  004517                                                                  BR      15$
     63 004506  104657  000021                MOV     U.NSEC(R5),R0   ; GET NUMBER OF SECTORS WRITTEN
     64 004510  115407                        INC     R0              ; SKIP THIS SECTOR
     65 004511  100657  000021                MOV     R0,U.NSEC(R5)   ; SAVE
     66 004513  104207  000001                MOV     #SETUP,R0       ; SETUP NEXT MODULE
     67 004515  104051                        MOV     R5,R1           ; DEFERRED CALL
     68 004516  005223                        BR      17$             ; EXIT
     69 004517  104651  000021        15$:    MOV     U.NSEC(R5),R1   ; GET SECTORS WRITTEN
     70 004521  105651  000024                ADD     U.CSEC(R5),R1   ; ADD TO TOTAL SECTORS WRITTEN
     71 004523  100651  000024                MOV     R1,U.CSEC(R5)   ; SAVE
     72 004525  104651  000021                MOV     U.NSEC(R5),R1   ; GET SECTORS WRITTEN
     73 004527  105651  000053                ADD     U.CBN(R5),R1    ; ADD TO LO ORDER CURRENT BN
     74 004531  100651  000053                MOV     R1,U CBN(R5)    ; SAVE
     75 004533  044541                        BCC     2$              ; IF NO CARRY, BRANCH
     76 004534  104651  000054                MOV     U.CBN+1(R5),R1  ; GET HI ORDER CURRENT BN
     77 004536  115401                        INC     R1              ; PROPOGATE CARRY
     78 004537  100651  000054                MOV     R1,U.CBN+1(R5)  ; SAVE
     79 004541  114001              2$:       CLR     R1              ; SET UP FOR NO SECTORS WTRITTEN
     80 004542  100651  000021                MOV     R1,U.NSEC(R5)   ; SAVE
     81 004544  104651  000061                MOV     U.RWER(R5),R1   ; GET WRITE ERROR
     82 004546  106201  000003                CMP     #3,R1           ; SEE IF REVECTOR
     83 004550  014607                        BEQ     1$              ; IF SO, BRANCH
     84 004551  106201  000002                CMP     #2,R1           ; SEE IF FAILURE IN DRIVE
     85 004553  054573                        BNE     6$              ; IF NOT, BRANCH
     86 004554                                SOFTER  6               ; REPORT
        004554  104200  000562  001107                                  MOV     #ER6,HRQ.04
        004557  104202  147646                                          MOV     #6!ERSOFT+4000.,R2
        004561  104020  001105                                          MOV     R2,HRQ.02
        004563  104200  004563  001104                                  MOV     #.,HRQ.01
        004566  104200  060013  001103                                  MOV     #ERRMES,HRQ.RQ
     87 004571  0242:3                        CALL    GOINIT
     88 004572  005124                        BR      12$             ; BRANCH
     89 004573  106201  000004        6$:     CMP     #4,R1           ; SEE IF HEADER COMPARE FAILURE
     90 004575  054721                        BNE     8$              ; IF NOT, BRANCH
     91 004576  104651  000046                MOV     U.PARM(R5),R1   ; GET UNIT PARAMETERS
     92 004600  104143                        MOV     (R4),R3         ; GET SUBUNIT PARAMETERS
     93 004601                                ASSUME  S.PARM,0
     94 004601  102203  020000                BIT     #DCYLS,R3       ; SEE IF USING DIAGNOSTIC CYLINDERS
     95 004603  054613                        BNE     7$              ; IF SO, BRANCH
     96 004604  102201  000200                BIT     #RBNBN,R1       ; SEE IF WRITING A RBN
     97 004606  054613                        BNE     7$              ; IF SO, BRANCH
     98 004607  104207  000022        1$:     MOV     #REVCT,R0       ; REVECTOR NEXT MODULE
     99 004611  114001                        CLR     R1              ; IMMIDATE CALL
    100 004612  005223                        BR      17$             ; EXIT
    101 004613                        7$:     HARDER  5               ; REPORT HEADER COMPARE FAILURE
        004613  104200  000365  001107                                  MOV     #ER5,HRQ.04
        004616  104202  107645                                          MOV     #5!ERHARD+4000.,R2
        004620  104020  001105                                          MOV     R2,HRQ.02
        004622  104200  004622  001104                                  MOV     #.,HRQ.01
        004625  104200  060014  001103                                  MOV     #ERRMC,HRQ.RQ
    102 004630                                ERRORC  <S.LETR(R4),RW.LOW(R0),RW.HI(R0)>
        004630  104640  000005  001110                                  MOV     S.LETR(R4),HRQ.05
        004633  104670  000002  001111                                  MOV     RW.LOW(R0),HRQ.06
```

```
                004636  104670  000003  001112                                  MOV     RW.HI(R0),HRQ.07
103 004641                                          ERRORC  <R1,#RBNTXT,U.RBN(R5)>
                004641  104010  001113                                          MOV     R1,HRQ.08
                004643  104200  005475  001114                                  MOV     #RBNTXT,HRQ.09
                004646  104650  000055  001115                                  MOV     U.RBN(R5),HRQ.10
104 004651                                          ERRORC  <U.RBN+1(R5),RW.ANG(R0)>
                004651  104650  000056  001116                                  MOV     U.RBN+1(R5),HRQ.11
                004654  104670  000006  001117                                  MOV     RW.ANG(R0),HRQ.12
105 004657                                          ERRORC  <RW.CMD(R0),U.CGRP(R5),U.CCYL(R5),U.CCYL+1(R5)>
                004657  104670  000004  001120                                  MOV     RW.CMD(R0),HRQ.13
                004662  104650  000066  001121                                  MOV     U.CGRP(R5),HRQ.14
                004665  104650  000064  001122                                  MOV     U.CCYL(R5),HRQ.15
                004670  104650  000065  001123                                  MOV     U.CCYL+1(R5),HRQ.16
106 004673                                          ERRORC  <U.LGRP(R5),U.LCYL(R5),U.LCYL+1(R5)>
                004673  104650  000071  001124                                  MOV     U.LGRP(R5),HRQ.17
                004676  104650  000067  001125                                  MOV     U.LCYL(R5),HRQ.18
                004701  104650  000070  001126                                  MOV     U.LCYL+1(R5),HRQ.19
107 004704                                          ENDERR  0
                004704  114000  002230                                          CLR     ERRPOS          ; CLEAR THE POSITION
108 004706  103201  000200                 BIC     #RBNBN,R1       ; NO LONGER HANDLING AN RBN
109 004710  100651  000046                 MOV     R1,U.PARM(R5)   ; SAVE
110 004712  104201  000001                 MOV     #1,R1           ; ONLY 1 SECTOR WRITTEN
111 004714  100651  000021                 MOV     R1,U.NSEC(R5)   ; SAVE
112 004716  104207  000001                 MOV     #SETUP,R0       ; SETUP NEXT MODULE
113 004720  005224                         BR      13$             ; EXIT AND REPORT
114 004721  106201  000153         8$:     CMP     #153,R1         ; SEE IF POSITIONER ERROR
115 004723  054765                         BNE     22$             ; IF NOT, BRANCH
116 004724                                 REPSFT  ,,SEEK          ; REPORT SEEK ERROR
                004724  114000  001105                                          CLR     HRQ.02
                004726  114000  001106                                          CLR     HRQ.03
                004730  104200  000001  001107                                  MOV     #1,HRQ.04
                004733  100467                                                          MOV R0,-(SP)
                004734  104657  000063                                          MOV     U.UNUM(R5),R0
                004736  105657  000050                                          ADD     U.SUBU(R5),R0
                004740  104070  001104                                          MOV     R0,HRQ.01
                004742  104207  060007                                          MOV     #T4SOFT,R0
                004744  021053                                                  CALL    HOSTRQ
                004745  104267                                                          MOV (SP)+,R0
117 004746                                 SOFTER  44              ; POSITIONER ERROR
                004746  104200  003132  001107                                  MOV     #ER44,HRQ.04
                004751  104202  147714                                          MOV     #44!ERSOFT+4000.,R2
                004753  104020  001105                                          MOV     R2,HRQ.02
                004755  104200  004755  001104                                  MOV     #.,HRQ.01
                004760  104200  060013  001103                                  MOV     #ERRMES,HRQ.RQ
118 004763  024235                         CALL    GORCLB          ; RECALIBRATE NEEDED
119 004764  005124                         BR      12$             ; BRANCH
120 004765  106201  000213         22$:    CMP     #213,R1         ; SEE IF READ/WRITE FAILURE
121 004767  055006                         BNE     23$             ; IF NOT, BRANCH
122 004770                                 SOFTER  48              ; READ/WRITE READY FAILURE
                004770  104200  003246  001107                                  MOV     #ER48,HRQ.04
                004773  104202  147720                                          MOV     #48!ERSOFT+4000.,R2
                004775  104020  001105                                          MOV     R2,HRQ.02
                004777  104200  004777  001104                                  MOV     #.,HRQ.01
                005002  104200  060013  001103                                  MOV     #ERRMES,HRQ.RQ
123 005005  005124                         BR      12$             ; ERROR EXIT
124 005006  106201  000253         23$:    CMP     #253,R1         ; SEE IF DRIVE DATA OR CLOCK TIMEOUT
125 005010  055030                         BNE     21$             ; IF NOT, BRANCH
```

```
    126 005011                                          SOFTER  47              ; DRIVE CLOCK TIMEOUT
        005011  104200  003217  001107                                                  MOV     #ER47,HRQ.04
        005014  104202  147717                                                          MOV     #47!ERSOFT+4000.,R2
        005016  104020  001105                                                          MOV     R2,HRQ.02
        005020  104200  005020  001104                                                  MOV     #.,HRQ.01
        005023  104200  060013  001103                                                  MOV     #ERRMES,HRQ.RQ
    127 005026  024243                                  CALL    GOINIT          ; INIT THE DRIVE TO RECOVER
    128 005027  005124                                  BR      12$             ; ERROR EXIT
    129 005030  106201  000313          21$:            CMP     #313,R1         ; SEE IF RECIEVER READY TIMEOUT
    130 005032  055052                                  BNE     24$             ; IF NOT, BRANCH
    131 005033                                          SOFTER  49              ; RECEIVER READY FAILURE
        005033  104200  003273  001107                                                  MOV     #ER49,HRQ.04
        005036  104202  147721                                                          MOV     #49!ERSOFT+4000.,R2
        005040  104020  001105                                                          MOV     R2,HRQ.02
        005042  104200  005042  001104                                                  MOV     #.,HRQ.01
        005045  104200  060013  001103                                                  MOV     #ERRMES,HRQ.RQ
    132 005050  024243                                  CALL    GOINIT          ; INIT THE DRIVE TO RECOVER
    133 005051  005124                                  BR      12$
    134 005052  106201  000413          24$:            CMP     #413,R1         ; SEE IF RTDS ERROR DURING WRITE
    135 005054  055073                                  BNE     25$             ; IF NOT, BRANCH
    136 005055                                          SOFTER  63              ; REPORT
        005055  104200  004120  001107                                                  MOV     #ER63,HRQ.04
        005060  104202  147737                                                          MOV     #63!ERSOFT+4000.,R2
        005062  104020  001105                                                          MOV     R2,HRQ.02
        005064  104200  005064  001104                                                  MOV     #.,HRQ.01
        005067  104200  060013  001103                                                  MOV     #ERRMES,HRQ.RQ
    137 005072  005124                                  BR      12$             ; BRANCH
    138 005073                          25$:            HARDER  68,<#SER36,R1>          ; UNKNOWN ERROR CODE
        005073  104200  004200  001107                                                  MOV     #ER68,HRQ.04
        005076  104200  004245  001110                                                  MOV     #SER36,HRQ.05
        005101  104010  001111                                                          MOV     R1,HRQ.06
        005103  104202  107744                                                          MOV     #68!ERHARD+4000.,R2
        005105  104020  001105                                                          MOV     R2,HRQ.02
        005107  104200  005107  001104                                                  MOV     #.,HRQ.01
        005112  104200  060014  001103                                                  MOV     #ERRMC,HRQ.RQ
    139 005115                                          ENDERR
        005115  104200  000051  001112                                                  MOV     #SER22,HRQ.07
        005120  104200  000010  002230                                                  MOV     #7+1,ERRPOS     ; SET THE POSITION
    140 005123  005215                                  BR      14$
    141 005124                          12$:            CERROR  5,#SER19
        005124  104200  000470  001110                                                  MOV     #SER19.HRQ.05
    142 005127                                          ERRORC  <U.RWTO(R5),S.LETR(R4),RW.LOW(R0),RW.HI(R0)>
        005127  104650  000012  001111                                                  MOV     U.RWTO(R5),HRQ.06
        005132  104640  000005  001112                                                  MOV     S.LETR(R4),HRQ.07
        005135  104670  000002  001113                                                  MOV     RW.LOW(R0),HRQ.08
        005140  104670  000003  001114                                                  MOV     RW.HI(R0),HRQ.09
    143 005143                                          ERRORC  <U.PARM(R5),#RBNTXT,U.RBN(R5)>
        005143  104650  000046  001115                                                  MOV     U.PARM(R5),HRQ.10
        005146  104200  005475  001116                                                  MOV     #RBNTXT,HRQ.11
        005151  104650  000055  001117                                                  MOV     U.RBN(R5),HRQ.12
    144 005154                                          ERRORC  <U.RBN+1(R5),RW.ANG(R0)>
        005154  104650  000056  001120                                                  MOV     U.RBN+1(R5),HRQ.13
        005157  104670  000006  001121                                                  MOV     RW.ANG(R0),HRQ.14
    145 005162                                          ERRORC  <RW.CMD(R0),U.CGRP(R5),U.CCYL(R5),U.CCYL+1(R5)>
        005162  104670  000004  001122                                                  MOV     RW.CMD(R0),HRQ.15
        005165  104650  000066  001123                                                  MOV     U.CGRP(R5),HRQ.16
        005170  104650  000064  001124                                                  MOV     U.CCYL(R5),HRQ.17
```

```
     005173  104650  000065  001125                                                  MOV     U.CCYL+1(R5),HRQ.18
146  005176                          ERRORC  <U.LGRP(R5),U.LCYL(R5),U.LCYL+1(R5)>
     005176  104650  000071  001126                                                  MOV     U.LGRP(R5),HRQ.19
     005201  104650  000067  001127                                                  MOV     U.LCYL(R5),HRQ.20
     005204  104650  000070  001130                                                  MOV     U.LCYL+1(R5),HRQ.21
147  005207                          ENDERR
     005207  104200  000051  001131                                                  MOV     #SER22,HRQ.22
     005212  104200  000027  002230                                                  MOV     #22+1,ERRPOS    ; SET THE POSITION
148  005215  024227                 14$:    CALL    GOSEEK          ; SEEK REQUIRED FOR ERROR RECOVERY
149  005216  104201  000001                 MOV     #1,R1           ; DEFERRED CALL
150  005220  100651  000022                 MOV     R1,U.MSEC(R5)   ; ONLY TRY TO WRITE ONE SECTOR
151  005222  005224                 13$:    BR      13$             ; BRANCH
152  005223  114002                 1?$:    CLR     R2              ; NO ERRORS
153  005224  003231                 ?3$:    BR      JMPRET          ; RETURN TO CALLING PROGRAM
154                                          .DSABL  LSB
155                                          .IF     LE,BUFARA-.
156                                 BUFARA   =       .+1
157                                          .ENDC
158                                          .IF     LE,MAXADR-.
159                                 MAXADR   =       .+1
160                                          .ENDC
```

```
                                        .SBTTL  ********** OVERLAY MODULE READ
                                                DMOVLY  R,AREA0
005225  000105                                    .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;
                                        ;
                                        ;
000012                                  READ    =       AFTWRT+1
                                        ;
                                        ;       READ READS A BLOCK FROM THE DEVICE
                                        ;
                                                .ENABL  LSB
004422  024212                                  CALL    DSABLE          ; DISABLE ERROR RECOVERY
004423  104657  000012                          MOV     U.RWTO(R5),R0   ; MOVE READ/WRITE TIMEOUT VALUE TO R0
004425  115407                                  INC     R0              ; INCREMENT COUNT
004426  100657  000012                          MOV     R0,U.RWTO(R5)   ; SAVE READ/WRITE TIMEOUT
004430  014540                                  BEQ     5$              ; IF ZERO, FIRST TIME -- BRANCH
004431  106207  000002                          CMP     #2,R0           ; SEE IF ATTEMPTED MAX TIMES
004433                                          BCS     11$             ; IF SO, BRANCH
004433  044435                                                                  BCC     .+2
004434  004446                                                                  BR      11$
004435  104141                                  MOV     (R4),R1         ; GET SUBUNIT PARAMETERS
004436                                          ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
004436  102201  001000                          BIT     #RTRIES,R1      ; SEE IF RETRIES ENABLED
004440  054540                                  BNE     5$              ; IF SO, BRANCH
004441  104651  000046                          MOV     U.PARM(R5),R1   ; GET UNIT PARAMETERS
004443  102201  000400                          BIT     #REVEC,R1       ; SEE IF READING RCT
004445  054540                                  BNE     5$              ; IF SO, GO FOR RETRIES
004446                                  11$:    HARDER  26,<S.LETR(R4),U.CBN(R5),U.CBN+1(R5)>
004446  104200  002371  001107                                                MOV     #ER26,HRQ.04
004451  104640  000005  001110                                                MOV     S.LETR(R4),HRQ.05
004454  104650  000053  001111                                                MOV     U.CBN(R5),HRQ.06
004457  104650  000054  001112                                                MOV     U.CBN+1(R5),HRQ.07
004462  104202  107672                                                        MOV     #26!ERHARD+4000.,R2
004464  104020  001105                                                        MOV     R2,HRQ.02
004466  104200  004466  001104                                                MOV     #.,HRQ.01
004471  104200  060014  001103                                                MOV     #ERRMC,HRQ.RQ
004474                                  ERRORC  <U.PARM(R5),#RBNTXT,U.RBN(R5),U.RBN+1(R5)>
004474  104650  000046  001113                                                MOV     U.PARM(R5),HRQ.08
004477  104200  005475  001114                                                MOV     #RBNTXT,HRQ.09
004502  104650  000055  001115                                                MOV     U.RBN(R5),HRQ.10
004505  104650  000056  001116                                                MOV     U.RBN+1(R5),HRQ.11
004510                                  ENDERR  0
004510  114000  002230                                                        CLR     ERRPOS          ; CLEAR THE POSITION
004512  104653  000046                  MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
004514  103203  000200                  BIC     #RBNBN,R3       ; IF HANDLING AN RBN, CLEAR IT
004516  100653  000046                  MOV     R3,U.PARM(R5)   ; SAVE
004520  104207  000001                  MOV     #1,R0           ; ONE SECTOR HANDLED
004522  100657  000021                  MOV     R0,U.NSEC(R5)   ; SAVE
004524  104207  000001                  MOV     #SETUP,R0       ; SETUP IS NEXT MODULE CALLED
004526  104051                          MOV     R5,R1           ; DELAYED CALL TO SETUP
004527  102203  000400                  BIT     #REVEC,R3       ; SEE IF READING RCT
004531  014545                          BEQ     8$              ; IF NOT, BRANCH
004532  104207  000022                  MOV     #REVCT,R0       ; REVECTOR NEXT MODULE
```

********** OVERLAY MODULE READ

```
     49 004534  104200  177777  002220          MOV     #-1,SCR2        ; MARK SECTOR AS BAD
     50 004537  004544                          BR      4$              ; EXIT
     51 004540  021777                  5$:     CALL    BULDUM          ; BUILD THE DUMMY SDI CONTROL BLOCK
     52 004541  024546                          CALL    RBLOCK          ; READ THE SECTOR
     53 004542  104207  000013                  MOV     #SECCHK,R0      ; SECCHK NEXT MODULE CALLED
     54 004544  114001                  4$:     CLR     R1              ; IMMIDATE CALL TO NEXT MODULE
     55 004545  003231                  2$:     BR      JMPRET          ; RETURN TO RDWRT MODULE
     56                                         .DSABL  LSB
```

```
                                        .SBTTL  RBLOCK - READ THE SECTORS
004546                                  RBLOCK:
                                        ;
                                        ;       READ A SECTOR FROM THE DEVICE
                                        ;
004546  104307  002232                  MOV     CHAINS,R0       ; POINT TO START OF CHAIN
004550  104652  000025                  MOV     U.MASK(R5),R2   ; R2 HAS SDI INTERCONNECT
004552  060012                          XFC     WAITSI          ; WAIT FOR SECTOR OR INDEX PULSE
004553  115001                          TST     R1              ; SEE IF ERROR OCCURRED
004554  014613                          BEQ     6$
004555                                  DEVFTL  42,<U.CGRP(R5),U.CCYL(R5),U.CCYL+1(R5)>
004555  104200  003071  001107                                          MOV     #ER42,HRQ.04
004560  104650  000066  001110                                          MOV     U.CGRP(R5),HRQ.05
004563  104650  000064  001111                                          MOV     U.CCYL(R5),HRQ.06
004566  104650  000065  001112                                          MOV     U.CCYL+1(R5),HRQ.07
004571  104202  047712                                                  MOV     #42!FTLDEV+4000.,R2
004573  104020  001105                                                  MOV     R2,HRQ.02
004575  104200  004575  001104                                          MOV     #.,HRQ.01
004600  104200  060014  001103                                          MOV     #ERRMC,HRQ.RQ
004603                                  ENDERR
004603  104200  000051  001113                                          MOV     #SER22,HRQ.08
004606  104200  000011  002230                                          MOV     #8+1,ERRPOS     ; SET THE POSITION
004611  024227                          CALL    GOSEEK
004612  004647                          BR      5$              ; EXIT
004613  060002                  6$:     XFC     XREAD           ; WRITE THE SECTOR(S)
004614  100651  000061                  MOV     R1,U.RWER(R5)   ; SAVE ERROR TYPE
004616  024650                          CALL    FNDRER          ; FIND READ ERROR (FILL IN CORRECT BN NUMBERS)
004617  115001                          TST     R1              ; SEE IF ANY ERRORS OCCURRED
004620  054646                          BNE     4$              ; IF SO, BRANCH
004621  104657  000012                  MOV     U.RWTO(R5),R0   ; SEE IF ANY RETRIES
004623  014646                          BEQ     4$              ; IF SO, BRANCH
004624                                  REPSFT  SOFT            ; REPORT SOFT ERROR
004624  104200  000001  001105                                          MOV     #1,HRQ.02
004627  114000  001106                                                  CLR     HRQ.03
004631  114000  001107                                                  CLR     HRQ.04
004633  100467                                                                  MOV R0,-(SP)
004634  104657  000063                                                  MOV     U.UNUM(R5),R0
004636  105657  000050                                                  ADD     U.SUBU(R5),R0
004640  104070  001104                                                  MOV     R0,HRQ.01
004642  104207  060007                                                  MOV     #T4SOFT,R0
004644  021053                                                          CALL    HOSTRQ
004645  104267                                                                  MOV (SP)+,R0
004646  114002                  4$:     CLR     R2              ; NO ERRORS
004647  000000                  5$:     RETURN                  ; RETURN TO CALLING PROGRAM:
```

```
                                .SBTTL  FNDRER - IF ERROR DURING READ, FIND IT'S POSITION IN THE CHAIN
004650                          FNDRER:
                                ;
                                ;       FIND THE FIRST BUFFER THAT IS NOT READ
                                ;
004650                          PUSH    <R1,R4,R5>      ; SAVE REGISTERS
004650  100461                                                          MOV R1,-(SP)
004651  100464                                                          MOV R4,-(SP)
004652  100465                                                          MOV R5,-(SP)
004653  104654  000053                MOV     U.CBN(R5),R4    ; GET LO ORDER BN
004655  104655  000054                MOV     U.CBN+1(R5),R5  ; GET HI BN
004657  104307  002232                MOV     CHAINS,R0       ; R0 POINTS TO FIRST BUFFER
004661  100674  000002        7$:     MOV     R4,RW.LOW(R0)   ; MOVE TO CHAIN
004663  100675  000003                MOV     R5,RW.HI(R0)    ; MOVE TO CHAIN
004665  104171                        MOV     (R0),R1         ; GET STATUS BITS
004666  102201  040000                BIT     #BUFFLG,R1      ; SEE IF THIS BUFFER HAS BEEN READ
004670  014703                        BEQ     8$              ; IF NOT, BRANCH
004671  115001                        TST     R1              ; SEE IF END-OF-LIST
004672  074703                        BMI     8$              ; IF LAST BUFFER, EXIT
004673                                ASSUME  RSTOP,100000    ; ASSUME READ STOP IS SIGN
004673  103201  170000                BIC     #^CHBHINB,R1    ; CLEAR UNUSED BITS
004675  104017                        MOV     R1,R0           ; MOVE TO R0
004676  105204  000001                ADD     #1,R4           ; ADD ONE TO LOW ORDER BN
004700  044661                        BCC     7$              ; IF NO CARRY, BRANCH
004701  115405                        INC     R5              ; PRPOGATE CARRY
004702  004661                        BR      7$              ; LOOP
004703                          8$:     POP     <R5,R4,R1>      ; RESTORE REGISTERS
004703  104265                                                          MOV (SP)+,R5
004704  104264                                                          MOV (SP)+,R4
004705  104261                                                          MOV (SP)+,R1
004706  000000                        RETURN                  ; RETURN TO CALLING PROGRAM
                                      .IF     LT,BUFARA-.
                                BUFARA  =       .
                                      .ENDC
                                      .IF     LE,MAXADR-.
                                MAXADR  =       .+1
                                      .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE SECCHK - CHECK THAT BUFFER FULL AND ECC
004707                                  DMOVLY  SC,AREA0
004707  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
000013                          SECCHK  =       READ+1
                                ;
                                ;       CHECK TO SEE IF THE SECTOR'S FULL, IF NOT, REPORT WHY (UNLESS
                                ;       REVECTOR, WHERE I'LL REVECTOR THE BLOCK)
                                ;       IF THE SECTOR'S FULL, CHECK ECC AND EDC, THEN CALL THE NEXT ROUTINE
                                ;       DEPENDING ON THE OPTIONS IN U.PARM (SELECTED BY USER)
                                ;
                                        .ENABL  LSB
004422  104307  002232                  MOV     CHAINS,R0       ; GET POINTER TO READ CHAIN
004424  104171                          MOV     (R0),R1         ; GET STATUS OF BUFFER
004425                                  ASSUME  RW.STA,0
004425  102201  040000                  BIT     #BUFFLG,R1      ; SEE IF BUFFER FULL
004427  055231                          BNE     4$              ; IF SO, BRANCH
004430  021725                          CALL    BLKCHK          ; SEE IF THIS IS A KNOWN BAD BLOCK
004431  045216                          BCC     3$              ; IF SO, BRANCH
004432  104651  000021                  MOV     U.NSEC(R5),R1   ; GET NUMBER OF SECTORS READ
004434  105651  000024                  ADD     U.CSEC(R5),R1   ; ADD NUMBER OF SECTORS PREVIOUSLY HANDLED
004436  100651  000024                  MOV     R1,U.CSEC(R5)   ; SAVE
004440  104651  000021                  MOV     U.NSEC(R5),R1   ; GET NUMBER OF SECTORS READ
004442  105641  000010                  ADD     S.MEGR(R4),R1   ; ADD TO MEGABITS READ
004444  100641  000010                  MOV     R1,S.MEGR(R4)   ; SAVE
004446  104651  000021                  MOV     U.NSEC(R5),R1   ; GET NUMBER OF SECTORS READ
004450  105651  000053                  ADD     U.CBN(R5),R1    ; ADJUST U.CBN TO SECTOR WITH ERROR
004452  100651  000053                  MOV     R1,U.CBN(R5)    ; SAVE
004454  044462                          BCC     1$              ; IF NO CARRY, BRANCH
004455  104651  000054                  MOV     U.CBN+1(R5),R1  ; GET CURRENT BN
004457  115401                          INC     R1              ; PROPOGATE CARRY
004460  100651  000054                  MOV     R1,U.CBN+1(R5)  ; SAVE
004462  114001                  1$:     CLR     R1              ; TO CLEAR NUMBER OF SECTORS HANDLED
004463  100651  000021                  MOV     R1,U.NSEC(R5)   ; ZERO SECTORS HANDLED
004465  104651  000061                  MOV     U.RWER(R5),R1   ; GET READ/WRITE ERROR
004467  106201  000003                  CMP     #3,R1           ; SEE IF REVECTORED BLOCK
004471  014533                          BEQ     2$              ; IF SO, BRANCH
004472  106201  000002                  CMP     #2,R1           ; SEE IF FAILURE IN DRIVE
004474  054514                          BNE     8$              ; IF NOT, BRANCH
004475                                  SOFTER  20              ; REPORT
004475  104200  002061  001107                                          MOV     #ER20,HRQ.04
004500  104202  147664                                                  MOV     #20!ERSOFT+4000.,R2
004502  104020  001105                                                  MOV     R2,HRQ.02
004504  104200  004504  001104                                          MOV     #.,HRQ.01
004507  104200  060013  001103                                          MOV     #ERRMES,HRQ.RQ
004512  024243                          CALL    GOINIT
004513  005117                          BR      12$             ; BRANCH
004514  106201  000004          8$:     CMP     #4,R1           ; SEE IF HEADER COMPARE FAILURE
004516  054652                          BNE     6$              ; IF NOT, BRANCH
004517  104653  000046                  MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
004521  104141                          MOV     (R4),R1         ; GET SUBUNIT PARAMETERS
```

```
                                        ASSUME  S.PARM,0
                                        BIT     #DCYLS,R1       ; SEE IF ON DIAGNOSTIC CYLINDERS
                                        BNE     9$              ; IF SO, BRANCH
                                        BIT     #REVEC,R3       ; SEE IF REVECTOR ENABLED
                                        BNE     5$              ; IF SO, BRANCH AROUND
                                        BIT     #RBNBN,R3       ; SEE IF READING RCT OR RBN
                                        BNE     9$              ; IF SO, REPORT ERROR
                                2$:     MOV     #REVCT,R0       ; REVECTOR NEXT MODULE
                                        BR      19$             ; EXIT
                                9$:     HARDER  19              ; REPORT HEADER COMPARE ERROR
004536  104200  001757  001107                                          MOV     #ER19,HRQ.04
004541  104202  107663                                                  MOV     #19!ERHARD+4000.,R2
004543  104020  001105                                                  MOV     R2,HRQ.02
004545  104200  004545  001104                                          MOV     #.,HRQ.01
004550  104200  060014  001103                                          MOV     #ERRMC,HRQ.RQ
                                        ERRORC  <S.LETR(R4),RW.LOW(R0),RW.HI(R0)>
004553  104640  000005  001110                                          MOV     S.LETR(R4),HRQ.05
004556  104670  000002  001111                                          MOV     RW.LOW(R0),HRQ.06
004561  104670  000003  001112                                          MOV     RW.HI(R0),HRQ.07
                                        ERRORC  <R3,#RBNTXT,U.RBN(R5)>
004564  104030  001113                                                  MOV     R3,HRQ.08
004566  104200  005475  001114                                          MOV     #RBNTXT,HRQ.09
004571  104650  000055  001115                                          MOV     U.RBN(R5),HRQ.10
                                        ERRORC  <U.RBN+1(R5),RW.ANG(R0)>
004574  104650  000056  001116                                          MOV     U.RBN+1(R5),HRQ.11
004577  104670  000006  001117                                          MOV     RW.ANG(R0),HRQ.12
                                        ERRORC  <RW.CMD(R0),U.CGRP(R5),U.CCYL(R5),U.CCYL+1(R5)>
004602  104670  000004  001120                                          MOV     RW.CMD(R0),HRQ.13
004605  104650  000066  001121                                          MOV     U.CGRP(R5),HRQ.14
004610  104650  000064  001122                                          MOV     U.CCYL(R5),HRQ.15
004613  104650  000065  001123                                          MOV     U.CCYL+1(R5),HRQ.16
                                        ERRORC  <U.LGRP(R5),U.LCYL(R5),U.LCYL+1(R5)>
004616  104650  000071  001124                                          MOV     U.LGRP(R5),HRQ.17
004621  104650  000067  001125                                          MOV     U.LCYL(R5),HRQ.18
004624  104650  000070  001126                                          MOV     U.LCYL+1(R5),HRQ.19
                                        ENDERR  0
004627  114000  002230                                                  CLR     ERRPOS          ; CLEAR THE POSITION
                                        BIC     #RBNBN,R3       ; IF HANDLING A RBN, CLEAR IT
                                        MOV     R3,U.PARM(R5)   ; SAVE
                                        MOV     #1,R1           ; ONLY ONE SECTOR HANDLED
                                        MOV     R1,U.NSEC(R5)   ; SAVE
                                        MOV     #SETUP,R0       ; SETUP NEXT MODULE
                                        BR      18$             ; BRANCH + REPORT, DELAYED CALL
                                5$:     MOV     #REVCT,R0       ; REVECTOR NEXT MODULE
                                        MOV     #-1,SCR2        ; SECTOR MARKED BAD
                                        BR      19$             ; EXIT, IMMIDATE CALL
                                6$:     CMP     #52,R1          ; SEE IF SERDES OVERRUN ERROR
                                        BNE     20$             ; IF NOT, BRANCH
                                        SOFTER  34              ; SERDES OVERRUN
004655  104200  002543  001107                                          MOV     #ER34,HRQ.04
004660  104202  147702                                                  MOV     #34!ERSOFT+4000.,R2
004662  104020  001105                                                  MOV     R2,HRQ.02
004664  104200  004664  001104                                          MOV     #.,HRQ.01
004667  104200  060013  001103                                          MOV     #ERRMES,HRQ.RQ
                                        BR      12$             ; REST OF ERROR MESSAGE
                                20$:    CMP     #150,R1         ; SEE IF DATA SYNC TIMEOUT
                                        BNE     22$             ; IF NOT, BRANCH
```

```
 90 004676                                          SOFTER  36          ; DATA SYNC TIMEOUT
    004676  104200  002615  001107                                              MOV     #ER36,HRQ.04
    004701  104202  147704                                                      MOV     #36!ERSOFT+4000.,R2
    004703  104020  001105                                                      MOV     R2,HRQ.02
    004705  104200  004705  001104                                              MOV     #.,HRQ.01
    004710  104200  060013  001103                                              MOV     #ERRMES,HRQ.RQ
 91 004713  005117                                          BR      12$         ; REST OF ERROR MESSAGE
 92 004714  106201  000153          22$:                    CMP     #153,R1     ; SEE IF POSITIONER ERROR
 93 004716  054760                                          BNE     23$         ; IF NOT, BRANCH
 94 004717                                          REPSFT  ,,SEEK              ; REPORT SEEK ERROR
    004717  114000  001105                                              CLR     HRQ.02
    004721  114000  001106                                              CLR     HRQ.03
    004723  104200  000001  001107                                      MOV     #1,HRQ.04
    004726  100467                                                              MOV R0,-(SP)
    004727  104657  000063                                              MOV     U.UNUM(R5),R0
    004731  105657  000050                                              ADD     U.SUBU(R5),R0
    004733  104070  001104                                              MOV     R0,HRQ.01
    004735  104207  060007                                              MOV     #T4SOFT,R0
    004737  021053                                                      CALL    HOSTRQ
    004740  104267                                                              MOV (SP)+,R0
 95 004741                                          SOFTER  45          ; REPORT POSITIONER ERROR
    004741  104200  003165  001107                                              MOV     #ER45,HRQ.04
    004744  104202  147715                                                      MOV     #45!ERSOFT+4000.,R2
    004746  104020  001105                                                      MOV     R2,HRQ.02
    004750  104200  004750  001104                                              MOV     #.,HRQ.01
    004753  104200  060013  001103                                              MOV     #ERRMES,HRQ.RQ
 96 004756  024235                                  CALL    GORCLB      ; RECALIBRATION REQUIRED FOR RECOVERY
 97 004757  005117                                  BR      12$         ; REST OF ERROR MESSAGE
 98 004760  106201  000213          23$:            CMP     #213,R1     ; SEE IF READ/WRITE READY FAILURE
 99 004762  055001                                  BNE     24$         ; IF NOT, BRANCH
100 004763                                          SOFTER  37          ; READ/WRITE READY FAILURE
    004763  104200  002637  001107                                              MOV     #ER37,HRQ.04
    004766  104202  147705                                                      MOV     #37!ERSOFT+4000.,R2
    004770  104020  001105                                                      MOV     R2,HRQ.02
    004772  104200  004772  001104                                              MOV     #.,HRQ.01
    004775  104200  060013  001103                                              MOV     #ERRMES,HRQ.RQ
101 005000  005117                                  BR      12$         ; REST OF ERROR MESSAGE
102 005001  106201  000253          24$:            CMP     #253,R1     ; SEE IF DATA DRIVE OR STATE CLOCK TIMEOUT
103 005003  055023                                  BNE     21$         ; IF NOT, BRANCH
104 005004                                          SOFTER  35          ; DRIVE CLOCK TIMEOUT
    005004  104200  002566  001107                                              MOV     #ER35,HRQ.04
    005007  104202  147703                                                      MOV     #35!ERSOFT+4000.,R2
    005011  104020  001105                                                      MOV     R2,HRQ.02
    005013  104200  005013  001104                                              MOV     #.,HRQ.01
    005016  104200  060013  001103                                              MOV     #ERRMES,HRQ.RQ
105 005021  024243                                  CALL    GOINIT      ; INIT THE DRIVE TO RECOVER
106 005022  005117                                  BR      12$         ; REST OF ERROR MESSAGE
107 005023  106201  000313          21$:            CMP     #313,R1     ; SEE IF RECIEVER READY TIMEOUT
108 005025  055045                                  BNE     25$         ; IF NOT, BRANCH
109 005026                                          SOFTER  38          ; RECIEVER READY FAILURE
    005026  104200  002663  001107                                              MOV     #ER38,HRQ.04
    005031  104202  147706                                                      MOV     #38!ERSOFT+4000.,R2
    005033  104020  001105                                                      MOV     R2,HRQ.02
    005035  104200  005035  001104                                              MOV     #.,HRQ.01
    005040  104200  060013  001103                                              MOV     #ERRMES,HRQ.RQ
110 005043  024243                                  CALL    GOINIT      ; INIT THE DRIVE TO RECOVER
111 005044  005117                                  BR      12$         ; REST OF ERROR MESSAGE
```

```
    112 005045  106201  000413          25$:   CMP     #413,R1          ; RTDS FAILURE DURING READ?
    113 005047  055066                         BNE     26$              ; IF NOT, BRANCH
    114 005050                                 SOFTER  64               ; REPORT
        005050  104200  004150  001107                                          MOV     #ER64,HRQ.04
        005053  104202  147740                                                  MOV     #64!ERSOFT+4000.,R2
        005055  104020  001105                                                  MOV     R2,HRQ.02
        005057  104200  005057  001104                                          MOV     #.,HRQ.01
        005062  104200  060013  001103                                          MOV     #ERRMES,HRQ.RQ
    115 005065  005117                         BR      12$              ; BRANCH
    116 005066                         26$:    HARDER  69,<#SER36,R1>   ; UNKNOWN ERROR NUMBER
        005066  104200  004223  001107                                          MOV     #ER69,HRQ.04
        005071  104200  004245  001110                                          MOV     #SER36,HRQ.05
        005074  104010  001111                                                  MOV     R1,HRQ.06
        005076  104202  107745                                                  MOV     #69!ERHARD+4000.,R2
        005100  104020  001105                                                  MOV     R2,HRQ.02
        005102  104200  005102  001104                                          MOV     #..,HRQ.01
        005105  104200  060014  001103                                          MOV     #ERRMC,HRQ.RQ
    117 005110                                 ENDERR
        005110  104200  000051  001112                                          MOV     #SER22,HRQ.07
        005113  104200  000010  002230                                          MOV     #7+1,ERRPOS      ; SET THE POSITION
    118 005116  005210                         BR      14$              ; REST OF ERROR MESSAGE
    119 005117                         12$:    CERROR  5,#SER19
        005117  104200  000470  001110                                          MOV     #SER19,HRQ.05
    120 005122                                 ERRORC  <U.RWTO(R5),S.LETR(R4),RW.LOW(R0),RW.HI(R0)>
        005122  104650  000012  001111                                          MOV     U.RWTO(R5),HRQ.06
        005125  104640  000005  001112                                          MOV     S.LETR(R4),HRQ.07
        005130  104670  000002  001113                                          MOV     RW.LOW(R0),HRQ.08
        005133  104670  000003  001114                                          MOV     RW.HI(R0),HRQ.09
    121 005136                                 ERRORC  <U.PARM(R5),#RBNTXT,U.RBN(R5)>
        005136  104650  000046  001115                                          MOV     U.PARM(R5),HRQ.10
        005141  104200  005475  001116                                          MOV     #RBNTXT,HRQ.11
        005144  104650  000055  001117                                          MOV     U.RBN(R5),HRQ.12
    122 005147                                 ERRORC  <U.RBN+1(R5),RW.ANG(R0)>
        005147  104650  000056  001120                                          MOV     U.RBN+1(R5),HRQ.13
        005152  104670  000006  001121                                          MOV     RW.ANG(R0),HRQ.14
    123 005155                                 ERRORC  <RW.CMD(R0),U.CGRP(R5),U.CCYL(R5),U.CCYL+1(R5)>
        005155  104670  000004  001122                                          MOV     RW.CMD(R0),HRQ.15
        005160  104650  000066  001123                                          MOV     U.CGRP(R5),HRQ.16
        005163  104650  000064  001124                                          MOV     U.CCYL(R5),HRQ.17
        005166  104650  000065  001125                                          MOV     U.CCYL+1(R5),HRQ.18
    124 005171                                 ERRORC  <U.LGRP(R5),U.LCYL(R5),U.LCYL+1(R5)>
        005171  104650  000071  001126                                          MOV     U.LGRP(R5),HRQ.19
        005174  104650  000067  001127                                          MOV     U.LCYL(R5),HRQ.20
        005177  104650  000070  001130                                          MOV     U.LCYL+1(R5),HRQ.21
    125 005202                                 ENDERR
        005202  104200  000051  001131                                          MOV     #SER22,HRQ.22
        005205  104200  000027  002230                                          MOV     #22+1,ERRPOS     ; SET THE POSITION
    126 005210  024227                 14$:    CALL    GOSEEK           ; SEEK REQUIRED FOR RECOVERY
    127 005211  104207  000001                 MOV     #1,R0            ; SET UP TO READ ONE SECTOR
    128 005213  100657  000022                 MOV     R0,U.MSEC(R5)    ; SAVE
    129 005215  005235                         BR      18$              ; EXIT
    130 005216  104641  000010         3$:     MOV     S.MEGR(R4),R1    ; GET NUMBER OF SECTORS READ
    131 005220  117401                         DEC     R1               ; READJUST TO ACTUAL SECTORS TESTED
    132 005221  100641  000010                 MOV     R1,S.MEGR(R4)    ; SAVE
    133 005223  104201  100000                 MOV     #EOC,R1          ; MOVE END-OF-CHAIN TO R1
    134 005225  100171                         MOV     R1,(R0)          ; FLAG THIS BUFFER AS END-OF-CHAIN (FOR LSTMOD)
    135 005226  104207  000021                 MOV     #LSTMOD,R0       ; LSTMOD NEXT MODULE
```

```
    136 005230  005233                        BR      19$             ; EXIT, NO ERRORS, IMMIDATE CALL
    137 005231  104207  000014          4$:   MOV     #CHKEDC,R0      ; CHKEDC IS NEXT MODULE
    138 005233  114002                  19$:  CLR     R2              ; NO ERRORS
    139 005234  114001                  17$:  CLR     R1              ; IMMIDATE CALL
    140 005235  003231                  18$:  BR      JMPRET          ; RETURN TO
    141                                       .DSABL  LSB
    142                                       .IF     LE,BUFARA-.
    143                                 BUFARA =      .+1
    144                                       .ENDC
    145                                       .IF     LE,MAXADR-.
    146                                 MAXADR =      .+1
    147                                       .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE CHKEDC - CHECK THAT ECC AND EDC ARE OK
005236                                  DMOVLY  ED,AREA0
005236  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
000014                          CHKEDC  =       SECCHK+1
                                ;
                                ;       CHECK THAT THERE IS NO ECC ERROR IN THIS BUFFER, IF NOT, CHECK THE
                                ;       EDC, IF THAT IS INCORRECT YOU HAVE A MAJOR ERROR...
                                ;       IF ECC AND EDC IS OK, GO TO DATA COMPARE (IF REQUESTED)
                                ;
                                        .ENABL  LSB
004422  104307  002232                  MOV     CHAINS,R0       ; R0 POINTS AT READ CHAIN LINK BEING TESTED
004424  104171                          MOV     (R0),R1         ; R1 IS BUFFER STATUS
004425  102201  010000                  BIT     #ECCFLG,R1      ; SEE IF ECC ERROR IN BUFFER
004427  014556                          BEQ     11$             ; IF NOT, CHECK EDC
004430  104653  000046                  MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
004432  102203  000400                  BIT     #REVEC,R3       ; SEE IF FINDING A REVECTOR
004434  054454                          BNE     2$              ; IF SO, BRANCH
004435  021725                          CALL    BLKCHK          ; SEE IF THIS IS A KNOWN BAD BLOCK
004436                                  BCS     5$              ; IF NOT, BRANCH
004436  044440                                                                  BCC     .+2
004437  004450                                                                  BR      5$
004440  104641  000010                  MOV     S.MEGR(R4),R1   ; GET NUMBER OF SECTORS READ
004442  117401                          DEC     R1              ; READJUST TO ACTUAL SECTORS HANDLED
004443  100641  000010                  MOV     R1,S.MEGR(R4)   ; SAVE
004445  104207  000021                  MOV     #LSTMOD,R0      ; LAST MODULE IS NEXT
004447  005005                          BR      19$             ; EXIT, NO ERRORS, IMMIDATE CALL
004450  104141                  5$:     MOV     (R4),R1         ; GET UNIT PARAMETERS
004451                                  ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
004451  102201  010000                  BIT     #ECCCHK,R1      ; SEE IF ECC CORRECTION IS REQUESTED
004453  014457                          BEQ     6$              ; IF NOT, BRANCH
004454  104207  000015          2$:     MOV     #CHKECC,R0      ; CHKECC NEXT MODULE
004456  005005                          BR      19$             ; EXIT, NO ERRORS, IMMIDATE CALL
004457                          6$:     SOFTER  7,<#SER21,U.RRTY(R5),U.ELEV(R5)>
004457  104200  000615  001107                                                  MOV     #ER7,HRQ.04
004462  104200  000661  001110                                                  MOV     #SER21,HRQ.05
004465  104650  000010  001111                                                  MOV     U.RRTY(R5),HRQ.06
004470  104650  000027  001112                                                  MOV     U.ELEV(R5),HRQ.07
004473  104202  147647                                                          MOV     #7!ERSOFT+4000.,R2
004475  104020  001105                                                          MOV     R2,HRQ.02
004477  104200  004477  001104                                                  MOV     #.,HRQ.01
004502  104200  060013  001103                                                  MOV     #ERRMES,HRQ.RQ
004505                                  ERRORC  <S.LETR(R4),RW.LOW(R0),RW.HI(R0)>
004505  104640  000005  001113                                                  MOV     S.LETR(R4),HRQ.08
004510  104670  000002  001114                                                  MOV     RW.LOW(R0),HRQ.09
004513  104670  000003  001115                                                  MOV     RW.HI(R0),HRQ.10
004516                                  ERRORC  <U.PARM(R5),#RBNTXT>
004516  104650  000046  001116                                                  MOV     U.PARM(R5),HRQ.11
004521  104200  005475  001117                                                  MOV     #RBNTXT,HRQ.12
004524                                  ERRORC  <U.RBN(R5),U.RBN+1(R5),RW.ANG(R0)>
004524  104650  000055  001120                                                  MOV     U.RBN(R5),HRQ.13
```

```
                                                                                MOV     U.RBN+1(R5),HRQ.14
                                                                                MOV     RW.ANG(R0),HRQ.15
48 004535                               ERRORC  <RW.CMD(R0),U.CGRP(R5),U.CCYL(R5)>
   004535 104670 000004 001123                                                  MOV     RW.CMD(R0),HRQ.16
   004540 104650 000066 001124                                                  MOV     U.CGRP(R5),HRQ.17
   004543 104650 000064 001125                                                  MOV     U.CCYL(R5),HRQ.18
49 004546                               ERRORC  U.CCYL+1(R5)
   004546 104650 000065 001126                                                  MOV     U.CCYL+1(R5),HRQ.19
50 004551                               ENDERR  0
   004551 114000 002230                                                         CLR     ERRPOS          ; CLEAR THE POSITION
51 004553 104651 000046                 MOV     U.PARM(R5),R1   ; GET UNIT PARAMETERS
52 004555 004751                        BR      10$             ; EXIT
53 004556 104677 000001         11$:    MOV     RW.BUF(R0),R0   ; R0 NOW POINTS AT BUFFER
54 004560 021146                        CALL    CMPEDC          ; COMPUTE THE EDC
55 004561 104651 000046                 MOV     U.PARM(R5),R1   ; GET UNIT PARAMETERS
56 004563 106?72 000400                 CMP     BF.EDC(R0),R2   ; SEE IF MATCH
57 004565 054641                        BNE     14$             ; IF NOT, BRANCH
58 004566 104657 000010                 MOV     U.RRTY(R5),R0   ; GET RETRIES
59 004570 054576                        BNE     23$             ; IF RETRIES WERE USED, REPORT SOFT ERROR
60 004571 104657 000027                 MOV     U.ELEV(R5),R0   ; GET ERROR LEVEL
61 004573 106657 000031                 CMP     U.MLEV(R5),R0   ; SEE IF ANY ERROR LEVELS WERE USED
62 004575 014620                        BEQ     22$             ; IF NOT, BRANCH
63 004576                       23$:    REPSFT  SOFT            ; REPORT SOFT ERROR
   004576 104200 000001 001105                                                  MOV     #1,HRQ.02
   004601 114000 001106                                                         CLR     HRQ.03
   004603 114000 001107                                                         CLR     HRQ.04
   004605 100467                                                                        MOV R0,-(SP)
   004606 104657 00006?                                                         MOV     U.UNUM(R5),R0
   004610 105657 00005?                                                         ADD     U.SUBU(R5),R0
   004612 104070 001104                                                         MOV     R0,HRQ.01
   004614 104207 060007                                                         MOV     #T4SOFT,R0
   004616 021053                                                                CALL    HOSTRQ
   004617 104267                                                                       MOV (SP)+,R0
64 004620 114000 002220         22$:    CLR     SCR2            ; FLAG AS GOOD BLOCK
65 004622 102201 000400                 BIT     #REVEC,R1       ; SEE IF REVECTOR IN PROGRESS
66 004624 014630                        BEQ     1$              ; IF NOT, BRANCH
67 004625 104207 000022                 MOV     #REVCT,R0       ; REVECTOR NEXT MODULE
68 004627 005005                        BR      19$             ; EXIT
69 004630 102201 000002         1$:     BIT     #DATCMP,R1      ; SEE IF DATA COMPARE REQUESTED
70 004632 054636                        BNE     12$             ; IF SO, BRANCH
71 004633 104207 000021                 MOV     #LSTMOD,R0      ; LSTMOD NEXT MODULE
72 004635 005005                        BR      19$             ; EXIT, NO ERRORS, IMMIDATE CALL
73 004636 104207 000020         12$:    MOV     #CMPDAT,R0      ; DATA COMPARE NEXT MODULE
74 004640 005005                        BR      19$             ; EXIT, NO ERRORS, IMMIDATE CALL
75 004641 104307 002232         14$:    MOV     CHAINS,R0
76 004643 104020 002220                 MOV     R2,SCR2         ; SAVE CALCULATED EDC
77 004645 104673 000001                 MOV     RW.BUF(R0),R3
78 004647                               SOFTER  24,<#SER21,U.RRTY(R5),U.ELEV(R5)>
   004647 104200 002275 001107                                                  MOV     #ER24,HRQ.04
   004652 104200 000661 001110                                                  MOV     #SER21,HRQ.05
   004655 104650 000010 001111                                                  MOV     U.RRTY(R5),HRQ.06
   004660 104650 000027 001112                                                  MOV     U.ELEV(R5),HRQ.07
   004663 104202 147670                                                         MOV     #24!ERSOFT+4000.,R2
   004665 104020 001105                                                         MOV     R2,HRQ.02
   004667 104200 004667 001104                                                  MOV     #.,HRQ.01
   004672 104200 060013 001103                                                  MOV     #ERRMES,HRQ.RQ
79 004675                               ERRORC  <S.LETR(R4),RW.LOW(R0),RW.HI(R0)>
```

```
                                                                        MOV     S.LETR(R4),HRQ.08
                                                                        MOV     RW.LOW(R0),HRQ.09
                                                                        MOV     RW.HI(R0),HRQ.10
 80 004706                                          ERRORC  <U.PARM(R5),#RBNTXT>
    004706  104650  000046  001116                                     MOV     U.PARM(R5),HRQ.11
    004711  104200  005475  001117                                     MOV     #RBNTXT,HRQ.12
 81 004714                                          ERRORC  <U.RBN(R5),U.RBN+1(R5),RW.ANG(R0)>
    004714  104650  000055  001120                                     MOV     U.RBN(R5),HRQ.13
    004717  104650  000056  001121                                     MOV     U.RBN+1(R5),HRQ.14
    004722  104670  000006  001122                                     MOV     RW.ANG(R0),HRQ.15
 82 004725                                          ERRORC  <RW.CMD(R0),U.CGRP(R5),U.CCYL(R5)>
    004725  104670  000004  001123                                     MOV     RW.CMD(R0),HRQ.16
    004730  104650  000066  001124                                     MOV     U.CGRP(R5),HRQ.17
    004733  104650  000064  001125                                     MOV     U.CCYL(R5),HRQ.18
 83 004736                                          ERRORC  <U.CCYL+1(R5),SCR2,BF.EDC(R3)>
    004736  104650  000065  001126                                     MOV     U.CCYL+1(R5),HRQ.19
    004741  104300  002220  001127                                     MOV     SCR2,HRQ.20
    004744  104630  000400  001130                                     MOV     BF.EDC(R3),HRQ.21
 84 004747                                          ENDERR  0
    004747  114000  002230                                            CLR     ERRPOS          ; CLEAR THE POSITION
 85 004751  104200  177777  002220  10$:    MOV     #-1,SCR2        ; FLAG AS BLOCK BAD
 86 004754  102201  000400              BIT     #REVEC,R1       ; SEE IF REVECTOR IN PROGRESS
 87 004756  054763                      BNE     13$             ; IF SO, RETRY
 88 004757  104147              15$:    MOV     (R4),R0         ; GET SUBUNIT PARAMETERS
 89 004760                              ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
 90 004760  102207  001000              BIT     #RTRIES,R0      ; SEE IF RETRIES ARE ENABLED
 91 004762  014766                      BEQ     16$             ; IF NOT, BRANCH
 92 004763  104207  000016              MOV     #ERCOV,R0       ; ERROR RECOVERY IS NEXT MODULE
 93 004765  005006                      BR      20$             ; EXIT, IMMIDATE CALL
 94 004766  103200  040000  001105  16$:    BIC     #C2HARD,HRQ.02  ; MAKE A HARD ERROR
 95 004771  104200  060014  001103          MOV     #ERRMC,HRQ.RQ   ; COUNT ERROR
 96 004774  102201  000002              BIT     #DATCMP,R1      ; SEE IF DATA COMPARE ENABLED
 97 004776  015002                      BEQ     17$             ; IF NOT, BRANCH
 98 004777  104207  000020              MOV     #CMPDAT,R0      ; DATA COMPARE IS NEXT MODULE
 99 005001  005006                      BR      20$             ; EXIT, IMMIDATE CALL
100 005002  104207  000021          17$:    MOV     #LSTMOD,R0      ; LSTMOD IS NEXT MODULE
101 005004  005006                      BR      20$             ; EXIT, IMMIDATE CALL
102 005005  114002              19$:    CLR     R2              ; NO ERRORS
103 005006  114001              20$:    CLR     R1              ; IMMIDATE CALL
104 005007  003231                      BR      JMPRET          ; RETURN TO
105                                     .DSABL  LSB
106                                     .IF     LE,BUFARA-.
107                             BUFARA  =       .+1
108                                     .ENDC
109
110                                     .IF     LE,MAXADR-.
111                             MAXADR  =       .+1
112                                     .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE CHKECC - CORRECT BUFFER READ USING ECC
 2 005010                               DMOVLY  CK,AREA0
   005010  000105                       .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
 3                              ;*******************************************************************************
 4                              ;*******************************************************************************
 5                              ;*******************************************************************************
 6                              ;*******************************************************************************
 7                              ;
 8                              ;
 9                              ;
10          000015              CHKECC  =       CHKEDC+1        ; DATA CHECK OVERLAY
11                              ;
12                              ;       SEE IF ECC ERROR WAS DECTECTED
13                              ;
21                                      .ENABL  LSB
22 004422  104307  002232               MOV     CHAINS,R0       ; R0 POINTS TO CHAIN
23 004424                               PUSH    R0              ; SAVE POINTER TO LINK
   004424  100467                                                                       MOV R0,-(SP)
24 004425  104200  177777  002220       MOV     #-1,SCR2        ; ASSUME BUFFER BAD
25 004430  060015                       XFC     ECC             ; APPLY ECC CORRECTION
26 004431  115001                       TST     R1              ; SEE IF CORRECTION WORKED
27 004432  014452                       BEQ     1$              ; IF SO, BRANCH
28 004433                               POP     R0              ; RESTORE POINTER TO LINK
   004433  104267                                                                       MOV (SP)+,R0
29 004434                               SOFTER  8
   004434  104200  000631  001107                                       MOV     #ER8,HRQ.04
   004437  104202  147650                                               MOV     #8!ERSOFT+4000.,R2
   004441  104020  001105                                               MOV     R2,HRQ.02
   004443  104200  004443  001104                                       MOV     #.,HRQ.01
   004446  104200  060013  001103                                       MOV     #ERRMES,HRQ.RQ
30 004451  004474                   BR      8$              ; BRANCH
31 004452  106657  000032       1$: CMP     U.ECCT(R5),R0   ; SEE IF CORRECTIONS = TO OR EXCEED THRESHOLD
32 004454  014456                   BEQ     6$              ; IF =, BRANCH
33 004455  034554                   BPL     2$              ; IF NOT, BRANCH
34 004456                       6$: POP     R0              ; RESTORE POINTER TO LINK
   004456  104267                                                                       MOV (SP)+,R0
35 004457                               SOFTER  9
   004457  104200  000745  001107                                       MOV     #ER9,HRQ.04
   004462  104202  147651                                               MOV     #9!ERSOFT+4000.,R2
   004464  104020  001105                                               MOV     R2,HRQ.02
   004466  104200  004466  001104                                       MOV     #.,HRQ.01
   004471  104200  060013  001103                                       MOV     #ERRMES,HRQ.RQ
36 004474                       8$: ERRORC  <#SER21,U.RRTY(R5),U.ELEV(R5)>
   004474  104200  000661  001110                                       MOV     #SER21,HRQ.05
   004477  104650  000010  001111                                       MOV     U.RRTY(R5),HRQ.06
   004502  104650  000027  001112                                       MOV     U.ELEV(R5),HRQ.07
37 004505                           ERRORC  <S.LETR(R4),RW.LOW(R0),RW.HI(R0)>
   004505  104640  000005  001113                                       MOV     S.LETR(R4),HRQ.08
   004510  104670  000002  001114                                       MOV     RW.LOW(R0),HRQ.09
   004513  104670  000003  001115                                       MOV     RW.HI(R0),HRQ.10
38 004516                           ERRORC  <U.PARM(R5),#RBNTXT>
   004516  104650  000046  001116                                       MOV     U.PARM(R5),HRQ.11
   004521  104200  005475  001117                                       MOV     #RBNTXT,HRQ.12
39 004524                           ERRORC  <U.RBN(R5),U.RBN+1(R5),RW.ANG(R0)>
   004524  104650  000055  001120                                       MOV     U.RBN(R5),HRQ.13
   004527  104650  000056  001121                                       MOV     U.RBN+1(R5),HRQ.14
   004532  104670  000006  001122                                       MOV     RW.ANG(R0),HRQ.15
```

```
     40 004535                                          ERRORC  <RW.CMD(R0),U.CGRP(R5),U.CCYL(R5)>
        004535  104670  000004  001123                                                  MOV     RW.CMD(R0),HRQ.16
        004540  104650  000066  001124                                                  MOV     U.CGRP(R5),HRQ.17
        004543  104650  000064  001125                                                  MOV     U.CCYL(R5),HRQ.18
     41 004546                                          ERRORC  U.CCYL+1(R5)
        004546  104650  000065  001126                                                  MOV     U.CCYL+1(R5),HRQ.19
     42 004551                                          ENDERR  0
        004551  114000  002230                                                          CLR     ERRPOS          ; CLEAR THE POSITION
     43 004553  004772                                  BR      11$             ; BRANCH
     44 004554                                  2$:     POP     R3              ; RESTORE POINTER TO BUFFER
        004554  104263                                                                  MOV (SP)+,R3
     45 004555  104637  000001                          MOV     RW.BUF(R3),R0   ; R0 POINTS TO BUFFER
     46 004557  021146                                  CALL    CMPEDC          ; COMPUTE EDC VALUE
     47 004560  106672  000400                          CMP     BF.EDC(R0),R2   ; SEE IF EDC VALUE MATCHES
     48 004562  054666                                  BNE     3$              ; IF NOT, BRANCH
     49 004563  104200  000001  001106                  MOV     #1,HRQ.03       ; REPORT ECC CORRECTION
     50 004566  104657  000010                          MOV     U.RRTY(R5),R0   ; GET TIMEOUT
     51 004570  054600                                  BNE     23$             ; IF RETRIES WERE USED, REPORT SOFT ERROR
     52 004571  104657  000027                          MOV     U.ELEV(R5),R0   ; GET ERROR LEVEL
     53 004573  114000  001105                          CLR     HRQ.02          ; ASSUME NO SOFT ERROR
     54 004575  106657  000031                          CMP     U.MLEV(R5),R0   ; SEE IF ANY ERROR LEVELS WERE USED
     55 004577  014602                                  BEQ     22$             ; IF NOT, BRANCH
     56 004600  115400  001105          23$:    INC     HRQ.02          ; REPORT SOFT ERROR
     57 004602                          22$:    PUSH    R0              ; SAVE R0
        004602  100467                                                                  MOV R0,-(SP)
     58 004603  104657  000063                          MOV     U.UNUM(R5),R0   ; GET STARTING SUBUNIT NUMBER
     59 004605  105657  000050                          ADD     U.SUBU(R5),R0   ; ADD OFFSET
     60 004607  104070  001104                          MOV     R0,HRQ.01       ; MOVE TO OUTPUT BUFFER
     61 004611  104207  060007                          MOV     #T4SOFT,R0      ; REPORT REQUEST
     62 004613  021053                                  CALL    HOSTRQ          ; SEND TO HOST
     63 004614                                  POP     R0              ; RESTORE R0
        004614  104267                                                                  MOV (SP)+,R0
     64 004615  114000  002220                          CLR     SCR2            ; BUFFER GOOD
     65 004617                                          MSSG    3,<S.LETR(R4),RW.LOW(R3),RW.HI(R3),RW.ANG(R3),RW.CMD(R3),U.CGRP(R5),U.CCYL(R
        004617  104200  005621  001105                                                  MOV     #MS3,HRQ.02
        004622  104640  000005  001106                                                  MOV     S.LETR(R4),HRQ.03
        004625  104630  000002  001107                                                  MOV     RW.LOW(R3),HRQ.04
        004630  104630  000003  001110                                                  MOV     RW.HI(R3),HRQ.05
        004633  104630  000006  001111                                                  MOV     RW.ANG(R3),HRQ.06
        004636  104630  000004  001112                                                  MOV     RW.CMD(R3),HRQ.07
        004641  104650  000066  001113                                                  MOV     U.CGRP(R5),HRQ.08
        004644  104650  000064  001114                                                  MOV     U.CCYL(R5),HRQ.09
        004647  104650  000065  001115                                                  MOV     U.CCYL+1(R5),HRQ.10
        004652  100467                                                                          MOV R0,-(SP)
        004653  104650  000063  001104                                                  MOV     U.UNUM(R5),HRQ.01
        004656  105650  000050  001104                                                  ADD     U.SUBU(R5),HRQ.01
        004661  104207  060015                                                          MOV     #MESSAG,R0
        004663  021053                                                                  CALL    HOSTRQ
        004664  104267                                                                          MOV (SP)+,R0
     66 004665  005015                                  BR      5$              ; EXIT
     67 004666  104020  002230          3$:     MOV     R2,ERRPOS       ; SAVE THE COMPUTED EDC
     68 004670                                          SOFTER  10,<#SER21,U.RRTY(R5),U.ELEV(R5)>
        004670  104200  000770  001107                                                  MOV     #ER10,HRQ.04
        004673  104200  000661  001110                                                  MOV     #SER21,HRQ.05
        004676  104650  000010  001111                                                  MOV     U.RRTY(R5),HRQ.06
        004701  104650  000027  001112                                                  MOV     U.ELEV(R5),HRQ.07
        004704  104202  147652                                                          MOV     #10!ERSOFT+4000.,R2
```

```
                                                                        MOV     R2,HRQ.02
                                                                        MOV     #.,HRQ.01
                                                                        MOV     #ERRMES,HRQ.RQ
 69 004716                                      ERRORC  <S.LETR(R4),RW.LOW(R3),RW.HI(R3)>
    004716  104640  000005  001113                                      MOV     S.LETR(R4),HRQ.08
    004721  104630  000002  001114                                      MOV     RW.LOW(R3),HRQ.09
    004724  104630  000003  001115                                      MOV     RW.HI(R3),HRQ.10
 70 004727                                      ERRORC  <U.PARM(R5),#RBNTXT>
    004727  104650  000046  001116                                      MOV     U.PARM(R5),HRQ.11
    004732  104200  005475  001117                                      MOV     #RBNTXT,HRQ.12
 71 004735                                      ERRORC  <U.RBN(R5),U.RBN+1(R5),RW.ANG(R3)>
    004735  104650  000055  001120                                      MOV     U.RBN(R5),HRQ.13
    004740  104650  000056  001121                                      MOV     U.RBN+1(R5),HRQ.14
    004743  104630  000006  001122                                      MOV     RW.ANG(R3),HRQ.15
 72 004746                                      ERRORC  <RW.CMD(R3),U.CGRP(R5),U.CCYL(R5)>
    004746  104630  000004  001123                                      MOV     RW.CMD(R3),HRQ.16
    004751  104650  000066  001124                                      MOV     U.CGRP(R5),HRQ.17
    004754  104650  000064  001125                                      MOV     U.CCYL(R5),HRQ.18
 73 004757                                      ERRORC  <U.CCYL+1(R5),ERRPOS,BF.EDC(R0)>        ; REPORT ERROR
    004757  104650  000065  001126                                      MOV     U.CCYL+1(R5),HRQ.19
    004762  104300  002230  001127                                      MOV     ERRPOS,HRQ.20
    004765  104670  000400  001130                                      MOV     BF.EDC(R0),HRQ.21
 74 004770                                      ENDERR  0
    004770  114000  002230                                              CLR     ERRPOS          ; CLEAR THE POSITION
 75 004772  104143                  11$:        MOV     (R4),R3         ; GET SUBUNIT PARAMETERS
 76 004773                                      ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
 77 004773  102203  001000                      BIT     #RTRIES,R3      ; SEE IF RETRIES ARE ENABLED
 78 004775  055012                              BNE     4$              ; IF SO, BRANCH
 79 004776  104653  000046                      MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
 80 005000  102203  000400                      BIT     #REVEC,R3       ; SEE IF REVECTOR IN PROGRESS
 81 005002  055012                              BNE     4$              ; IF SO, BRANCH
 82 005003  103200  040000  001105              BIC     #C2HARD,HRQ.02  ; MAKE SOFT ERROR A HARD ERROR
 83 005006  104200  060014  001103              MOV     #ERRMC,HRQ.RQ   ; COUNT ERROR
 84 005011  005026                              BR      10$             ; BRANCH
 85 005012  104207  000016          4$:         MOV     #ERCOV,R0       ; ERROR RECOVERY IS NEXT MODULE
 86 005014  005036                              BR      7$              ; BRANCH
 87 005015  114002                  5$:         CLR     R2              ; NO ERRORS
 88 005016  104653  000046                      MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
 89 005020  102203  000400                      BIT     #REVEC,R3       ; SEE IF FINDING A REVECTOR
 90 005022  015026                              BEQ     10$             ; IF NOT, BRANCH
 91 005023  104207  000022                      MOV     #REVCT,R0       ; REVECTOR NEXT MODULE
 92 005025  005036                              BR      7$              ; BRANCH
 93 005026  102203  000002          10$:        BIT     #DATCMP,R3      ; SEE IF DATA COMPARE IS REQUESTED
 94 005030  015034                              BEQ     9$              ; IF NOT, BRANCH
 95 005031  104207  000020                      MOV     #CMPDAT,R0      ; DATA COMPARE IS NEXT MODULE
 96 005033  005036                              BR      7$              ; BRANCH
 97 005034  104207  000021          9$:         MOV     #LSTMOD,R0      ; GO TO LAST MODULE
 98 005036  114001                  7$:         CLR     R1              ; IMMIDATE CALL TO NEXT MODULE
 99 005037  003231                              BR      JMPRET          ; RETURN
100                                             .DSABL  LSB
101                                             .IF     LE,BUFARA-.
102                                 BUFARA      =       .+1
103                                             .ENDC
104                                             .IF     LE,MAXADR-.
105                                 MAXADR      =       .+1
106                                             .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE ERCOV - DATA ERROR RETRIES AND LEVELS
005040                                  DMOVLY  EC,AREA0
005040  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
000016                          ERCOV   =       CHKECC+1
                                ;
                                ;       ERROR RECOVERY AND RETRIES
                                ;
                                        .ENABL  LSB
004422  024203                          CALL    ENABLE          ; ENABLE ERROR RECOVERY
004423  104657  000010                  MOV     U.RRTY(R5),R0   ; GET NUMBER OF RETRIES ALLREADY ATTEMPTED
004425  054464                          BNE     1$              ; IF NOT ZERO (FIRST TIME) BRANCH
004426  104651  000021                  MOV     U.NSEC(R5),R1   ; GET NUMBER OF SECTORS READ
004430  105651  000024                  ADD     U.CSEC(R5),R1   ; ADD NUMBER OF SECTORS PREVIOUSLY HANDLED
004432  100651  000024                  MOV     R1,U.CSEC(R5)   ; SAVE
004434  104651  000021                  MOV     U.NSEC(R5),R1   ; GET NUMBER OF SECTORS READ
004436  105641  000010                  ADD     S.MEGR(R4),R1   ; ADD TO MEGABITS WRITTEN
004440  100641  000010                  MOV     R1,S.MEGR(R4)   ; SAVE
004442  104651  000021                  MOV     U.NSEC(R5),R1   ; GET NUMBER OF SECTORS READ
004444  114003                          CLR     R3              ; TO SETUP VALUES
004445  100653  000021                  MOV     R3,U.NSEC(R5)   ; NO SECTORS READ
004447  115403                          INC     R3              ; MAKE R3 ONE
004450  100653  000022                  MOV     R3,U.MSEC(R5)   ; ONLY READ ONE SECTOR
004452  105651  000053                  ADD     U.CBN(R5),R1    ; ADJUST U.CBN TO SECTOR WITH ERROR
004454  100651  000053                  MOV     R1,U.CBN(R5)    ; SAVE
004456  044464                          BCC     1$              ; IF NO CARRY, BRANCH
004457  104651  000054                  MOV     U.CBN+1(R5),R1  ; GET CURRENT BN
004461  115401                          INC     R1              ; PROPOGATE CARRY
004462  100651  000054                  MOV     R1,U.CBN+1(R5)  ; SAVE
004464  106657  000030          1$:     CMP     U.RTRY(R5),R0   ; COMPARE MAXIMUM COUNT WITH RETRIES ATTEMPTED
004466  014502                          BEQ     NLEV            ; IF RETRIES EXHAUSTED, BRANCH
004467  115407                          INC     R0              ; INCREMENT RETRY COUNT
004470  100657  000010                  MOV     R0,U.RRTY(R5)   ; SAVE TIMEOUT
004472  104207  177777                  MOV     #-1,R0          ; START READ RETRIES AT ZERO
004474  100657  000012                  MOV     R0,U.RWTO(R5)   ; SAVE
004476  104207  000007                  MOV     #BUILDP,R0      ; BUILDP IS NEXT MODULE
004500  104051                          MOV     R5,R1           ; DELAYED CALL TO READ
004501  004513                          BR      EREXT           ; BRANCH
004502  104207  000017          NLEV:   MOV     #NEWLEV,R0      ; NEW LEVEL OF ERROR RECOVERY WILL BE TRIED
004504  104653  000047                  MOV     U.RCOV(R5),R3   ; GET ERROR RECOVERY PARAMETERS
004506  101203  010000                  BIS     #NXTLEV,R3      ; GO TO NEXT LEVEL
004510  100653  000047                  MOV     R3,U.RCOV(R5)   ; SAVE
004512  114001                          CLR     R1              ; IMMIDATE CALL TO NEXT MODULE
004513  114002          EREXT:  CLR     R2              ; NO ERRORS
004514  003231                          BR      JMPRET          ; RETURN TO                51
                                        .IF     LE,BUFARA-.
                                BUFARA  =       .+1
                                        .ENDC
                                        .IF     LE,MAXADR-.
                                MAXADR  =       .+1
```

```
     57                                             .ENDC
```

```
                                        .SBTTL  ********** OVERLAY MODULE NEWLEV - SEND DRIVE ERROR LEVEL
 2 004515                               .       DMOVLY  NL,AREA0
   004515  000105                       .       .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
 3                                      ;*******************************************************************************
 4                                      ;*******************************************************************************
 5                                      ;*******************************************************************************
 6                                      ;*******************************************************************************
 7                                      ;
 8                                      ;
 9                                      ;
10          000017                      NEWLEV  =       ERCOV+1
11                                      ;
12                                      ;       INITIATE A NEW LEVEL OF ERROR RECOVERY
13                                      ;
21                                              .ENABL  LSB
22 004422  104653  000047                       MOV     U.RCOV(R5),R3   ; GET RECOVERY WORD
23 004424  102203  010000                       BIT     #NXTLEV,R3      ; GO TO NEXT LEVEL?
24 004426  054433                               BNE     3$              ; IF SO, BRANCH
25 004427  104657  000027                       MOV     U.ELEV(R5),R0   ; GET CURRENT ERROR RECOVERY LEVEL
26 004431  115407                               INC     R0              ; GO UP ONE LEVEL
27 004432  004540                               BR      LEVNZR
28 004433  104657  000027              3$:      MOV     U.ELEV(R5),R0   ; GET CURRENT ERROR RECOVERY LEVEL
29 004435  054540                               BNE     LEVNZR          ; IF NON-ZERO, BRANCH
30 004436  024212                               CALL    DSABLE          ; DISABLE ERROR RECOVERY
31 004437                                       HARDER  11,<S.LETR(R4),U.CBN(R5),U.CBN+1(R5)>
   004437  104200  001044  001107                                               MOV     #ER11,HRQ.04
   004442  104640  000005  001110                                               MOV     S.LETR(R4),HRQ.05
   004445  104650  000053  001111                                               MOV     U.CBN(R5),HRQ.06
   004450  104650  000054  001112                                               MOV     U.CBN+1(R5),HRQ.07
   004453  104202  J7653                                                        MOV     #11!ERHARD+4000.,R2
   004455  104020  001105                                                       MOV     R2,HRQ.02
   004457  104200  004457  001104                                               MOV     #.,HRQ.01
   004462  104200  060014  001103                                               MOV     #ERRMC,HRQ.RQ
32 004465                                       ERRORC  <U.PARM(R5),#RBNTXT,U.RBN(R5),U.RBN+1(R5),U.CGRP(R5)>
   004465  104650  000046  001113                                               MOV     U.PARM(R5),HRQ.08
   004470  104200  005475  001114                                               MOV     #RBNTXT,HRQ.09
   004473  104650  000055  001115                                               MOV     U.RBN(R5),HRQ.10
   004476  104650  000056  001116                                               MOV     U.RBN+1(R5),HRQ.11
   004501  104650  000066  001117                                               MOV     U.CGRP(R5),HRQ.12
33 004504                                       ERRORC  <U.CCYL(R5),U.CCYL+1(R5)>
   004504  104650  000064  001120                                               MOV     U.CCYL(R5),HRQ.13
   004507  104650  000065  001121                                               MOV     U.CCYL+1(R5),HRQ.14
34 004512                                       ENDERR  0
   004512  114000  002230                                                       CLR     ERRPOS          ; CLEAR THE POSITION
35 004514  104653  000046                       MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
36 004516  104200  177777  002220               MOV     #-1,SCR2        ; FLAG AS BAD SECTOR
37 004521  102203  000400                       BIT     #REVEC,R3       ; SEE IF FINDING A REVECTORED SECTOR
38 004523  014527                               BEQ     1$              ; IF NOT, BRANCH
39 004524  104207  000022                       MOV     #REVCT,R0       ; GO TO REVCT ROUTINE NEXT
40 004526  004600                               BR      NEWEXT          ; BRANCH TO EXIT
41 004527  102203  000002              1$:      BIT     #DATCMP,R3      ; SEE IF DATA COMPARE REQUESTED
42 004531  014535                               BEQ     2$              ; IF NOT, BRANCH
43 004532  104207  000020                       MOV     #CMPDAT,R0      ; COMPARE DATA NEXT MODULE
44 004534  004600                               BR      NEWEXT          ; EXIT
45 004535  104207  000021              2$:      MOV     #LSTMOD,R0      ; LAST MODULE NEXT MODULE
46 004537  004600                               BR      NEWEXT          ; EXIT
47                                              .DSABL  LSB
```

```
     48 004540  104070  001677           LEVNZR: MOV     R0,ERRLEV       ; MOVE LEVEL TO SDI COMMAND
     49 004542  104203  001643                   MOV     #CR.ERR,R3      ; POINT TO ERROR RECOVERY COMMAND
     50 004544  021173                           CAL     TALK            ; SEND SDI INTERCHANGE
     51 004545  115002                           TST     R2              ; SEE IF AN ERROR OCCURRED
     52 004546  014553                           BEQ     1$              ; IF NOT, BRANCH
     53 004547                                   CERROR  5,#SER4         ; REPORT ERROR
        004547  104200  004731  001110                                                MOV     #SER4,HRQ.05
     54 004552  004601                           BR      SNDEX           ; BRANCH
     55 004553  104657  000047           1$:     MOV     U.RCOV(R5),R0   ; GET RECOVERY PARAMETERS
     56 004555  102207  010000                   BIT     #NXTLEV,R0      ; DID THIS GO TO THE NEXT LEVEL?
     57 004557  014573                           BEQ     2$              ; IF NOT, BRANCH
     58 004560  103207  010000                   BIC     #NXTLEV,R0      ; CLEAR NEXT LEVEL BIT
     59 004562  101207  020000                   BIS     #LEVUSD,R0      ; FLAG AS LEVELS USED
     60 004564  100657  000047                   MOV     R0,U.RCOV(R5)   ; SAVE
     61 004566  104657  000027                   MOV     U.ELEV(R5),R0   ; GET ERROR RECOVERY LEVEL
     62 004570  117407                           DEC     R0              ; DECREMENT
     63 004571  100657  000027                   MOV     R0,U.ELEV(R5)   ; SAVE
     64 004573  104207  000007           2$:     MOV     #BUILDP,R0      ; BUILDP IS NEXT MODULE CALLED
     65 004575  114002                           CLR     R2              ; NO ERRORS
     66 004576  100652  000010                   MOV     R2,U.RRTY(R5)   ; START RETRIES AT 0
     67 004600  114001                   NEWEXT: CLR     R1              ; IMMIDATE CALL TO NEXT MODULE
     68 004601  003231                   SNDEX:  BR      JMPRET          ; RETURN TO         R
     69                                           .IF     LE,BUFARA-.
     70                                   BUFARA  =       .+1
     71                                           .ENDC
     72                                           .IF     LE,MAXADR-.
     73                                   MAXADR  =       .+1
     74                                           .ENDC
```

```
                                        .SBTTL  ********** OVERLAY MODULE CMPDAT - DATA COMPARISON ON READ BUFFER(S)
                                                DMOVLY  CD,AREA0
004602  000105                                  .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;
                                        ;
                                        ;
000020                                  CMPDAT  =       NEWLEV+1
                                        ;
                                        ;       CMPDAT COMPARES THE PATTERN WITH THAT READ FROM THE SECTOR
                                        ;
                                                .ENABL  LSB
004422                                  PUSH    <R4,R5>         ; SAVE R4,R5
004422  100464                                                                  MOV R4,-(SP)
004423  100465                                                                  MOV R5,-(SP)
004424  104307  002232                         MOV     CHAINS,R0       ; R0 POINTS TO LINK
004426  104677  000001                         MOV     RW.BUF(R0),R0   ; R0 POINTS TO BUFFER
004430  104172                                 MOV     (R0),R2         ; R2 HAS PATTERN NUMBER (IN EACH NIBBLE)
004431  103202  177760                         BIC     #LBLONB,R2      ; CLEAR UNUSED BITS
004433  054440                                 BNE     1$              ; IF NOT PATTERN 16, BRANCH
004434  104200  000020  002221                 MOV     #16.,PNUM       ; REPORT PATTERN 16
004437  004442                                 BR      2$              ; BRANCH
004440  104020  002221                 1$:     MOV     R2,PNUM         ; SAVE PATTERN NUMBER
004442  104021                         2$:     MOV     R2,R1           ; BUILD PATTERN NUMBER WORD IN R1
004443  110702                                 SWAB    R2              ; MOVE PATTERN NUMBER TO HI BYTE
004444  101021                                 BIS     R2,R1           ; SET HI BITS
004445  110202                                 ROL     R2              ; ROTATE TO HI NIBBLE
004446  110202                                 ROL     R2
004447  110202                                 ROL     R2
004450  110202                                 ROL     R2
004451  103202  007777                         BIC     #HBHINB,R2      ; CLEAR UNUSED BITS
004453  101021                                 BIS     R2,R1           ; SET BITS
004454  110702                                 SWAB    R2              ; MOVE TO LO BYTE
004455  101021                                 BIS     R2,R1           ; SET BITS
004456  106271                                 CMP     (R0)+,R1        ; SEE IF REDUNDANT PATTERN OK
004457  054523                                 BNE     FWRD            ; IF NOT, BRANCH
004460  104302  002221                         MOV     PNUM,R2         ; RESTORE R2
004462  103202  177760                         BIC     #LBLONB,R2      ; MAP PATTERN 16 TO PATTERN 0
004464  104622  002236                         MOV     PATPTR(R2),R2   ; POINT TO PATTERN
004466  115402                                 INC     R2              ; SKIP EDC
004467  104201  000001                         MOV     #1,R1           ; R1 HAS OFFSET INTO BUFFER
                                                .DSABL  LSB
004471  104024                         XOPLP0: MOV     R2,R4           ; R4 POINTS TO LENGTH OF PATTERN
004472  104243                                 MOV     (R4)+,R3        ; R3 CONTAINS LENGTH OF PATTERN
004473  106203  000001                         CMP     #1,R3           ; SEE IF PATTERN IS 1 WORD LONG
004475  014510                                 BEQ     ONEPAX          ; IF SO, BRANCH
004476  104245                         XOPLP1: MOV     (R4)+,R5        ; R5 GETS 1 WORD OF THE DATA PATTERN
004477  106275                                 CMP     (R0)+,R5        ; COMPARE PATTERN WORD TO SECTOR AREA
004500  054524                                 BNE     CMPERR          ; IF NOMATCH, BRANCH
004501  115401                                 INC     R1              ; INCREMENT OFFSET
004502  106201  000400                         CMP     #SCTWRD+1,R1    ; SEE IF ENTIRE BUFFER COMPARED
004504  014517                                 BEQ     CSCEXT          ; IF ALL WORDS COMPARED, BRANCH
004505  117403                                 DEC     R3              ; DECREMENT COUNT OF WORDS IN PATTERN
004506  054476                                 BNE     XOPLP1          ; IF DATA PATTERN UNFINISHED, BRANCH
```

```
     62 004507  004471                          BR      XOPLP0          ; BRANCH
     63 004510  104145                ONEPAX:   MOV     (R4),R5         ; GET 1 WORD OF DATA PATTERN
     64 004511  106275                XOPLP2:   CMP     (R0)+,R5        ; COMPARE PATTERN TO SECTOR READ
     65 004512  054524                          BNE     CMPERR          ; IF COMPARE ERROR, BRANCH
     66 004513  115401                          INC     R1              ; INCREMENT NUMBER OF WORDS TO COMPARE
     67 004514  106201  000400                  CMP     #SCTWRD+1,R1    ; SEE IF ALL WORDS COMPARED
     68 004516  054511                          BNE     XOFLP2          ; IF INCOMPLETE, BRANCH
     69 004517  114002                CSCEXT:   CLR     R2              ; NO ERRORS
     70 004520                                  POP     <R5,R4>         ; RESTORE R5,R4
        004520  104265                                                                  MOV (SP)+,R5
        004521  104264                                                                  MOV (SP)+,R4
     71 004522  004716                          BR      CMXEX           ; BRANCH
     72 004523  114001                FWRD:     CLR     R1              ; ZERO OFFSET
     73 004524                        CMPERR:   POP     <R5,R4>         ; RESTORE R5,R4
        004524  104265                                                                  MOV (SP)+,R5
        004525  104264                                                                  MOV (SP)+,R4
     74 004526                                  PUSH    R0              ; SAVE POINTER TO BUFFER
        004526  100467                                                                  MOV R0,-(SP)
     75 004527  104037                          MOV     R3,R0           ; MOVE POINTER TO READ CHAIN TO R0
     76 004530  021725                          CALL    BLKCHK          ; SEE IF THIS IS A KNOWN BAD BLOCK
     77 004531                                  POP     R0              ; RESTORE R0
        004531  104267                                                                  MOV (SP)+,R0
     78 004532  044716                          BCC     CMXEX           ; IF KNOWN BAD BLOCK, EXIT
     79 004533  104303  002232                  MOV     CHAINS,R3       ; R3 POINTS TO LINK
     80 004535                                  HARDER  12
        004535  104200  001116  001107                                         MOV     #ER12,HRQ.04
        004540  104202  107654                                                  MOV     #12!ERHARD+4000.,R2
        004542  104020  001105                                                  MOV     R2,HRQ.02
        004544  104200  004544  001104                                         MOV     #.,HRQ.01
        004547  104200  060014  001103                                         MOV     #ERRMC,HRQ.RQ
     81 004552                                  CERROR  5,#SER24        ; ECC/EDC HAD DETECTED ERROR
        004552  104200  001331  001110                                         MOV     #SER24,HRQ.05
     82 004555  115000  002220                  TST     SCR2            ; SEE IF ERROR DETECTED
     83 004557  054563                          BNE     4$              ; IF SO, BRANCH
     84 004560                                  CERROR  5,#SER25        ; ECC/EDC DID NOT DETECT ERROR
        004560  104200  001357  001110                                         MOV     #SER25,HRQ.05
     85 004563                        4$:       ERRORC  <S.LETR(R4),RW.LOW(R3),RW.HI(R3)>
        004563  104640  000005  001111                                         MOV     S.LETR(R4),HRQ.06
        004566  104630  000002  001112                                         MOV     RW.LOW(R3),HRQ.07
        004571  104630  000003  001113                                         MOV     RW.HI(R3),HRQ.08
     86 004574                                  ERRORC  <U.PARM(R5),#RBNTXT>
        004574  104650  000046  001114                                         MOV     U.PARM(R5),HRQ.09
        004577  104200  005475  001115                                         MOV     #RBNTXT,HRQ.10
     87 004602                                  ERRORC  <U.RBN(R5),U.RBN+1(R5),RW.ANG(R3)>
        004602  104650  000055  001116                                         MOV     U.RBN(R5),HRQ.11
        004605  104650  000056  001117                                         MOV     U.RBN+1(R5),HRQ.12
        004610  104630  000006  001120                                         MOV     RW.ANG(R3),HRQ.13
     88 004613                                  ERRORC  <RW.CMD(R3),U.CGRP(R5),U.CCYL(R5),U.CCYL+1(R5)>
        004613  104630  000004  001121                                         MOV     RW.CMD(R3),HRQ.14
        004616  104650  000066  001122                                         MOV     U.CGRP(R5),HRQ.15
        004621  104650  000064  001123                                         MOV     U.CCYL(R5),HRQ.16
        004624  104650  000065  001124                                         MOV     U.CCYL+1(R5),HRQ.17
     89 004627  106201  000366                  CMP     #246.,R1        ; SEE IF IN LAST 9 WORDS
     90 004631  034642                          BPL     1$              ; IF NOT, BRANCH
     91 004632  104013                          MOV     R1,R3           ; TO COMPUTE OFFSET OF ERROR
     92 004633  107203  000367                  SUB     #247.,R3        ; R3 HAS UNADJUSTED ERROR POSITION
     93 004635  105203  000003                  ADD     #3,R3           ; R3 HAS ADJUSTED ERROR POSITION
```

```
     94 004637  107037                          SUB     R3,R0           ; ADJUST POINTER
     95 004640  117407                          DEC     R0              ; COMPARE AUTO-INCREMENTED, SO ADJUST
     96 004641  004655                          BR      3$              ; PRINT ERROR
     97 004642  106201  000004          1$:     CMP     #4,R1           ; SEE IF IN FIRST 4 WORDS
     98 004644  044652                          BCC     2$              ; IF SO, BRANCH
     99 004645  107207  000004                  SUB     #4,R0           ; LOOK BACK FOUR WORDS
    100 004647  104203  000003                  MOV     #3,R3           ; OFFSET OF ERROR IN PRINTOUT IS 3
    101 004651  004655                          BR      3$              ; BRANCH
    102 004652  107017                  2$:     SUB     R1,R0           ; BACK UP POINTER
    103 004653  104013                          MOV     R1,R3           ; R3 IS OFFSET OF ERROR IN PRINTOUT
    104 004654  117407                          DEC     R0              ; TO POINT TO START OF BUFFER
    105 004655                          3$:     ERRORC  <PNUM,R1,R3,(R0)+,(R0)+,(R0)+,(R0)+,(R0)+,(R0)+,(R0)+>
        004655  104300  002221  001125                                          MOV     PNUM,HRQ.18
        004660  104010  001126                                                  MOV     R1,HRQ.19
        004662  104030  001127                                                  MOV     R3,HRQ.20
        004664  104270  001130                                                  MOV     (R0)+,HRQ.21
        004666  104270  001131                                                  MOV     (R0)+,HRQ.22
        004670  104270  001132                                                  MOV     (R0)+,HRQ.23
        004672  104270  001133                                                  MOV     (R0)+,HRQ.24
        004674  104270  001134                                                  MOV     (R0)+,HRQ.25
        004676  104270  001135                                                  MOV     (R0)+,HRQ.26
        004700  104270  001136                                                  MOV     (R0)+,HRQ.27
        004702  104270  001137                                                  MOV     (R0)+,HRQ.28
    106 004704                          ERRORC  <(R0)+,(R0)+,(R0)+,(R0)+>
        004704  104270  001140                                                  MOV     (R0)+,HRQ.29
        004706  104270  001141                                                  MOV     (R0)+,HRQ.30
        004710  104270  001142                                                  MOV     (R0)+,HRQ.31
        004712  104270  001143                                                  MOV     (R0)+,HRQ.32
    107 004714                          ENDERR  0
        004714  114000  002230                                                  CLR     ERRPOS          ; CLEAR THE POSITION
    108 004716  104207  000021  CMXEX:  MOV     #LSTMOD,R0      ; LAST MODULE WILL BE CALLED
    109 004720  114001                  CLR     R1              ; CALL IMMIDATELY
    110 004721  003231                  BR      JMPRET          ; RETURN TO
    111                                 .IF     LE,BUFARA-.
    112                         BUFARA  =       .+1
    113                                 .ENDC
    114                                 .IF     LE,MAXADR-.
    115                         MAXADR  =       .+1
    116                                 .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE LSTMOD - CLEANUP MODULE BEFORE SETUP
                                        DMOVLY  LM,AREA0
004722  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
000021                          LSTMOD  =       CMPDAT+1
                                ;
                                ;       LOOP CONTROL FOR ECC, EDC AND DATA COMPARE MODULES.  ALSO LAST
                                ;       MODULE BEFORE SETUP (USUALLY)
                                ;
                                        .ENABL  LSB
004422  114002                          CLR     R2              ; ASSUME NO ERRORS
004423  104657  000021                  MOV     U.NSEC(R5),R0   ; GET NUMBER OF SECTORS HANDLED SO FAR
004425  115407                          INC     R0              ; INCREMENT COUNT
004426  100657  000021                  MOV     R0,U.NSEC(R5)   ; SAVE
004430  104701  175600                  MOV     @CHAINS,R1      ; SEE IF LAST BUFFER LAST READ
004432  074443                          BMI     1$              ; IF SO, BRANCH
004433                                  ASSUME  RW.STAT,0       ; ASSUME STATUS FIRST WORD
004433                                  ASSUME  RSTOP,100000    ; ASSUME READ STOP IS SIGN BIT
004433                                  ASSUME  EOC,100000
004433  103201  170000                  BIC     #^CHBHINB,R1    ; CLEAR UNUSED BITS
004435  104010  002232                  MOV     R1,CHAINS       ; CHAINS NOW POINTS TO NEXT LINK
004437  104207  000013                  MOV     #SECCHK,R0      ; SECTOR CHECK NEXT MODULE
004441  114001                          CLR     R1              ; IMMIDATE CALL
004442  004476                          BR      4$              ; EXIT
004443  104653  000046          1$:     MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
004445  103203  000200                  BIC     #RBNBN,R3       ; IF HANDLING AN RBN, CLEAR IT
004447  100653  000046                  MOV     R3,U.PARM(R5)   ; SAVE
004451  105647  000010                  ADD     S.MEGR(R4),R0   ; GET MEGABYTE COUNT
004453  106207  003642                  CMP     #1954.,R0       ; SEE IF ONE MEGABYTE TRANSFERED
004455  034471                          BPL     3$              ; IF NOT, BRANCH
004456  107207  003642                  SUB     #1954.,R0       ; ZERO COUNT
004460  104200  000001  001105          MOV     #1,HRQ.02       ; REPORT 1 MEGABYTE READ
004463  114000  001106                  CLR     HRQ.03          ; NO WRITE MEGABYTES REPORTED
004465  104202  060011                  MOV     #T4MXFR,R2      ; SET UP FOR MEGABIT REPORT
004467  104020  001103                  MOV     R2,HRQ.RQ       ; FLAG AS NON-ERROR
004471  100647  000010          3$:     MOV     R0,S.MEGR(R4)   ; SAVE COUNT
004473  104207  000001                  MOV     #SETUP,R0       ; SETUP IS NEXT MODULE CALLED
004475  104051                          MOV     R5,R1           ; DEFERRED CALL TO NEXT MODULE
004476  003231                  4$:     BR      JMPRET          ; RETURN TO          R
                                        .DSABL  LSB
                                        .IF     LE,BUFARA-.
                                BUFARA  =       .+1
                                        .ENDC
                                        .IF     LE,MAXADR-.
                                MAXADR  =       .+1
                                        .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE REVCT - REVECTORED SECTOR HANDLING
                                        DMOVLY  RV,AREA0
004477  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*' ***********************************************************************
                                ;*   ***********************************************************************
                                ;*  ************************************************************************
                                ;************************************************************************
                                ;
                                ;
                                ;
000022                          REVCT   =       LSTMOD+1        ; REVECTOR OVERLAY
                                        .ENABL  LSB
004422  104657  000046                  MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
004424  102207  000400                  BIT     #REVEC,R0       ; SEE IF REVECTOR ALLREADY IN PROGRESS
004426  054504                          BNE     1$              ; IF SO, BRANCH
                                .SBTTL  REVSUP - SETUP THE REVECTOR OPERATION (TO READ RCT)
                                ;REVSUP
                                ;
                                ;       INITILIZE ALL PARAMETERS FOR READING THE RCT TO FING THE RBN THAT
                                ;       A HEADER HAS BEEN REVECTORED TO
                                ;
004427  101207  000400                  BIS     #REVEC,R0       ; FLAG AS REVECTOR IN PROGRESS
004431  100657  000046                  MOV     R0,U.PARM(R5)   ; SAVE PARAMETERS
004433  104207  005526                  MOV     #RCTL$,R0       ; POINT OT RCT LBN STRING
004435  100647  000005                  MOV     R0,S.LETR(R4)   ; SAVE
004437  104657  000061                  MOV     U.RWER(R5),R0   ; GET READ/WRITE ERROR TYPE
004441  100657  000062                  MOV     R0,U.RVER(R5)   ; SAVE FOR REVECTOR INFORMATION
004443  104641  000007                  MOV     S.SCHR(R4),R1   ; R1 POINTS TO SUBUNIT CHARACTERISTICS
004445  114007                          CLR     R0              ; USE R0 TO INITILIZE VALUES
004446  100657  000055                  MOV     R0,U.RBN(R5     ; START WITH RBN ZERO
004450  100657  000056                  MOV     R0,U.RBN+1(R5)  ; START WITH RBN ZERO
004452  100657  000060                  MOV     R0,U.CCOP(R5)   ; ON ORIGINAL COPY OF RCT
004454  115407                          INC     R0              ; R0 IS NOW 1
004455  100657  000022                  MOV     R0,U.MSEC(R5)   ; ONLY READ 1 SECTOR AT A TIME
004457  115407                          INC     R0              ; R0 IS NOW 2
004460  105617  000012                  ADD     LBNHST(R1),R0   ; R0 POINTS TO 1ST REVECTOR INFORMATION SECTOR
004462  100657  000053                  MOV     R0,U.CBN(R5)    ; SAVE
004464  104617  000013                  MOV     LBNHST+1(R1),R0 ; R0 HAS HI FIRST REV INFO SECTOR
004466  044470                          BCC     5$              ; IF NO CARY, BRANCH
004467  115407                          INC     R0              ; PROPOGATE CARRY
004470  103207  170000          5$:     BIC     #^CHBHINB,R0    ; CLEAR SUBUNIT BITS
004472  100657  000054                  MOV     R0,U.CBN+1(R5)  ; SAVE
004474  114002                          CLR     R2              ; NO ERRORS
004475  104201  177777                  MOV     #-1,R1          ; START READ ATTEMPT RETRIES AT 0
004477  100651  000012                  MOV     R1,U.RWTO(R5)   ; SAVE
004501  104207  000023                  MOV     #SEEK,R0        ; SEEK IS NEXT MODULE
004503  004533                          BR      4$              ; EXIT
004504  104651  000024          1$:     MOV     U.CSEC(R5),R1   ; GET NUMBER OF SECTORS R/W SO FAR
004506  105651  000051                  ADD     U.MBN(R5),R1    ; ADD STARTING SECTOR OF OPERATION
004510  104010  002175                  MOV     R1,CURBN        ; MOVE TO TEMP STORAGE
004512  104651  000052                  MOV     U.MBN+1(R5),R1  ; GET HI STARTING SECTOR
004514  044516                          BCC     2$              ; IF NO CARRY, BRANCH
004515  115401                          INC     R1              ; PROPOGATE CARRY
004516  104010  002176          2$:     MOV     R1,CURBN+1      ; SAVE
004520  104301  002220                  MOV     SCR2,R1         ; SEE IF SECTOR READ IS OK
004522  014525                          BEQ     6$              ; IF SO, BRANCH
004523  024534                          CALL    REVSOK          ; FIND NEXT COPY TO READ
```

```
     64 004524  004533                        BR      4$              ; EXIT
     65 004525  024644              6$:       CALL    SEARCH          ; SEARCH THE SECTOR TO FIND THE LBN
     66 004526  115001                        TST     R1              ; SEE IF LBN FOUND
     67 004527  014533                        BEQ     4$              ; LBN FOUND, BRANCH AND READ RBN
     68 004530  115002                        TST     R2              ; SEE IF NULL FLAG FOUND (REVECTOR NOT FOUND)
     69 004531  054533                        BNE     4$              ; IF NOT, BRANCH
     70 004532  025003                        CALL    NXTRCT          ; READ NEXT RCT SECTOR
     71 004533  003231              4$:       BR      JMPRET          ; RETURN CONTROL TO         R
     72                                       .DSABL  LSB
```

REVSOK - SEE IF THE REVECTOR INFO SECTOR JUST READ IS OK

```
                                        .SBTTL  REVSOK - SEE IF THE REVECTOR INFO SECTOR JUST READ IS OK
004534                                  REVSOK:
                                        ;
                                        ;       GO TO THE NEXT RCT COPY
                                        ;
004534  104651  000060                  MO;     U.CCOP(R5),R1   ; GET NUMBER OF COPIES TRIED SO FAR
004536  115401                          INC     R1              ; TRY ANOTHER COPY
004537  106651  000057                  CMP     COPY(R5),R1     ; CHECK AGAINST MAX
004541  034613                          BPL     5$              ; IF ALL COPIES UNTRIED, BRANCH
004542                                  HARDER  40
004542  104200  002710  001107                                                  MOV     #ER40,HRQ.04
004545  104202  107710                                                          MOV     #40!ERHARD+4000.,R2
004547  104020  001105                                                          MOV     R2,HRQ.02
004551  104200  004551  001104                                                  MOV     #.,HRQ.01
004554  104200  060014  001103                                                  MOV     #ERRMC,HRQ.RQ
004557  104651  000062                  MOV     U.RVER(R5),R1   ; GET ORIGINAL ERROR TYPE
004561  106201  000003                  CMP     #3,R1           ; SEE IF REVECTORED BLOCK
004563  054570                          BNE     2$              ; IF NOT, BRANCH
004564                                  CERROR  5,#SER26        ; FLAG ERROR
004564  104200  003040  001110                                                  MOV     #SER26,HRQ.05
004567  004573                          BR      36$             ; EXIT
004570                          2$:     CERROR  5,#SER32        ; FLAG ERROR HEADER COMPARE ERROR
004570  104200  003054  001110                                                  MOV     #SER32,HRQ.05
004573                          36$:    ERRORC  <U.CBN(R5),U.CBN+1(R5),CURBN,CURBN+1>
004573  104650  000053  001111                                                  MOV     U.CBN(R5),HRQ.06
004576  104650  000054  001112                                                  MOV     U.CBN+1(R5),HRQ.07
004601  104300  002175  001113                                                  MOV     CURBN,HRQ.08
004604  104300  002176  001114                                                  MOV     CURBN+1,HRQ.09
004607                                  ENDERR  0
004607  114000  002230                                                          CLR     ERRPOS          ; CLEAR THE POSITION
004611  025134                          CALL    RVFAIL          ; CLEAR ALL BITS AND DO A FAIL EXIT
004612  004643                          BR      7$              ; EXIT
004613  100651  000060          5$:     MOV     R1,U.CCOP(R5)   ; SAVE COPY COUNT
004615  104647  000007                  MOV     S.SCHR(R4),R0   ; R0 POINTS TO SUBUNIT PARAMETERS
004617  104677  000014                  MOV     RCTCSZ(R0),R0   ; R0 IS RCT COPY SIZE
004621  105657  000053                  ADD     U.CBN(R5),R0    ; ADD CURRENT SECTOR
004623  100657  000053                  MOV     R0,U.CBN(R5)    ; SAVE
004625  044633                          BCC     6$              ; IF NO CARRY, BRANCH
004626  104657  000054                  MOV     U.CBN+1(R5),R0  ; GET HI CURRENT BLOCK NUMBER
004630  115407                          INC     R0              ; INCREMENT COUNT
004631  100657  000054                  MOV     R0,U.CBN+1(R5)  ; SAVE
004633  024203                  6$:     CALL    ENABLE
004634  104203  177777                  MOV     #-1,R3          ; START READ ATTEMPT RETRIES AT 0
004636  100653  000012                  MOV     R3,U.RWTO(R5)   ; SAVE
004640  104207  000023                  MOV     #SEEK,R0        ; SEEK IS NEXT MODULE
004642  114002                          CLR     R2              ; NO ERRORS
004643  000000                  7$:     RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  SEARCH - TRY TO FIND THE LBN IN THE RCT SECTOR JUST READ
004644                                  SEARCH:
                                        ;
                                        ;       SEARCH THE RCT SECTOR JUST READ TO FIND THE LBN OR THE NULL ENTRY
                                        ;
004644  104307  002232                          MOV     CHAINS,R0       ; R0 POINTS TO LINK (NODE) IN READ CHAIN
004646  104677  000001                          MOV     RW.BUF(R0),R0   ; R0 NOW POINTS TO BUFFER
004650  114001                                  CLR     R1              ; R1 IS RBN NUMBER OFFSET WITHIN THE SECTOR
004651  104672  000001                  1$:     MOV     1(R0),R2        ; GET RCT CODE
004653  074737                                  BMI     6$              ; IF NULL POINTER, ENTIRE TABLE EXHAUSTED, BRANCH
004654  102202  020000                          BIT     #020000,R2      ; SEE IF IT IS A USED RBN
004656  014715                                  BEQ     5$              ; IF NOT, BRANCH
004657  103202  170000                          BIC     #^CHBHINB,R2    ; CLEAR CODE
004661  106302  002176                          CMP     CURBN+1,R2      ; SEE IF HI ORDER MATCHES
004663  054715                                  BNE     5$              ; IF NOT, BRANCH
004664  106170  002175                          CMP     (R0),CURBN      ; SEE IF LO ORDER MATCHES
004666  054715                                  BNE     5$              ; IF NOT, BRANCH
004667  105651  000055                          ADD     U.RBN(R5),R1    ; ADD RUNNING RBN TO OFFSET
004671  100651  000055                          MOV     R1,U.RBN(R5)    ; SAVE
004673  044701                                  BCC     2$              ; IF NO CARRY, BRANCH
004674  104651  000056                          MOV     U.RBN+1(R5),R1  ; GET HI ORDER RBN
004676  115401                                  INC     R1              ; PROPOGATE CARRY
004677  100651  000056                          MOV     R1,U.RBN+1(R5)  ; SAVE
004701  104657  000046                  2$:     MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
004703  101207  000200                          BIS     #RBNBN,R0       ; FLAG ALL ROUTINES THAT THIS IS AN RBN
004705  100657  000046                          MOV     R0,U.PARM(R5)   ; SAVE
004707  025134                                  CALL    RVFAIL          ; FAIL EXIT DOES WHAT WE WANT, JUST CLEAR R1 AND R2
004710  114001                                  CLR     R1              ; FOUND IT
004711  114002                                  CLR     R2              ; NO ERRORS
004712  100652  000021                          MOV     R2,U.NSEC(R5)   ; SAVE
004714  005002                                  BR      7$              ; EXIT
004715  105207  000002                  5$:     ADD     #2,R0           ; POINT TO NEXT RBN RECORD
004717  115401                                  INC     R1              ; INCREMENT RBN OFFSET
004720  106201  000200                          CMP     #128.,R1        ; SEE IF ALL RECORDS TRIED
004722  054651                                  BNE     1$              ; IF NOT, BRANCH
004723  114002                                  CLR     R2              ; TO SIGNAL CALLING ROUTINE TO READ NEXT SECTOR
004724  105651  000055                          ADD     U.RBN(R5),R1    ; ADD OLD RBN TO 128
004726  100651  000055                          MOV     R1,U.RBN(R5)    ; SAVE
004730  045002                                  BCC     7$              ; IF NO CARRY, EXIT
004731  104651  000056                          MOV     U.RBN+1(R5),R1  ; GET HI ORDER RBN
004733  115401                                  INC     R1              ; PROPOGATE CARRY
004734  100651  000056                          MOV     R1,U.RBN+1(R5)  ; SAVE
004736  005002                                  BR      7$              ; EXIT
004737                                  6$:     HARDER  41
004737  104200  003000  001107                                                  MOV     #ER41,HRQ.04
004742  104202  107711                                                          MOV     #41!ERHARD+4000.,R2
004744  104020  001105                                                          MOV     R2,HRQ.02
004746  104200  004746  001104                                                  MOV     #.,HRQ.01
004751  104200  060014  001103                                                  MOV     #ERRMC,HRQ.RQ
004754  104651  000062                          MOV     U.RVER(R5),R1   ; GET ORIGINAL ERROR
004756  106201  000003                          CMP     #3,R1           ; SEE IF REVECTORED BLOCK
004760  054765                                  BNE     32$             ; IF NOT, BRANCH
004761                                          CERROR  5,#SER26        ; REVECTORED SECTOR
004761  104200  003040  001110                                                  MOV     #SER26,HRQ.05
004764  004770                                  BR      12$             ; EXIT
004765                                  32$:    CERROR  5,#SER32        ; REPORT HEADER COMPARE ERROR
004765  104200  003054  001110                                                  MOV     #SER32,HRQ.05
```

```
     51 004770                          12$:    ERRORC  <CURBN,CURBN+1>
        004770  104300  002175  001111                                    MOV     CURBN,HRQ.06
        004773  104300  002176  001112                                    MOV     CURBN+1,HRQ.07
     52 004776                                  ENDERR  0
        004776  114000  002230                                            CLR     ERRPOS          ; CLEAR THE POSITION
     53 005000  025134                  9$:     CALL    RVFAIL          ; CLEAR ALL BITS, FAIL EXIT
     54 005001  104051                          MOV     R5,R1           ; FLAG AS ERROR
     55 005002  000000                  7$:     RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  NXTRCT - GET THE NEXT RCT SECTOR TO SEARCH
005003                                  NXTRCT:
                                        ;
                                        ;       INCREMENT CBN BY 1 AFTER SUBTACTING THE NUNMBER OF COPIES*RCT SIZE
                                        ;       THAT WAS SEARCHED THROUGH ON THE LAST PASS
                                        ;
005003  104652  000053                          MOV     U.CBN(R5),R2    ; R2 HAS LO ORDER RCT BN
005005  104653  000054                          MOV     U.CBN+1(R5),R3  ; R3 HAS HI ORDER RCT BN
005007  104657  000060                          MOV     U.CCOP(R5),R0   ; GET NUMBER OF COPIES THAT HAVE BEEN READ
005011  015022                                  BEQ     3$              ; IF NO COPIES, BRANCH
005012  104641  000007                          MOV     S.SCHR(R4),R1   ; R1 POINTS TO SUBUNIT CHARACTERISTICS
005014  107612  000014                  1$:     SUB     RCTCSZ(R1),R2   ; SUBTRACT FROM LO ORDER BN
005016  045020                                  BCC     2$              ; IF NO BORROW, BRANCH
005017  117403                                  DEC     R3              ; PROPOGATE BORROW
005020  117407                          2$:     DEC     R0              ; DECREMENT COUNT
005021  055014                                  BNE     1$              ; IF NO CARRY, BRANCH
005022  105202  000001                  3$:     ADD     #1,R2           ; INCREMENT CBN BY 1
005024  045026                                  BCC     4$              ; IF NO CARRY, BRANCH
005025  115403                                  INC     R3              ; PROPOGATE CARRY
005026  100652  000053                  4$:     MOV     R2,U.CBN(R5)    ; SAVE LO ORDER
005030  100653  000054                          MOV     R3,U.CBN+1(R5)  ; SAVE HI ORDER
005032  104020  002175                          MOV     R2,CURBN        ; MOVE TO CALC AREA
005034  104030  002176                          MOV     R3,CURBN+1      ; MOVE TO CALC AREA
                                        ;                               NOTE R0 MUST BE ZERO FOR CALC SETUP
005036  022027                                  CALL    CALC            ; FIND CYL THAT NEXT SEC IS ON
005037  104647  000007                          MOV     S.SCHR(R4),R0   ; POINT TO SUBUNIT CHARACTERISTICS
005041  106670  000001  002202                  CMP     HICYL(R0),CYL+1 ; SEE IF IN XBN AREA
005044                                          BCS     5$              ; IF SO, BRANCH
005044  045046                                                                  BCC     .+2
005045  005054                                                                  BR      5$
005046  055123                                  BNE     6$              ; IF NOT, BRANCH
005047  106670  000000  002201                  CMP     LBNCYL(R0),CYL  ; SEE IF IN XBN AREA
005052  015054                                  BEQ     5$              ; IF SO, BRANCH
005053  045123                                  BCC     6$              ; IF NOT, BRANCH
005054  025134                          5$:     CALL    RVFAIL          ; REVECTOR FAILED
005055                                          HARDER  4               ; REPORT CORRUPTED RCT
005055  104200  000271  001107                                                  MOV     #ER4,HRQ.04
005060  104202  107644                                                          MOV     #4!ERHARD+4000.,R2
005062  104020  001105                                                          MOV     R2,HRQ.02
005064  104200  005064  001104                                                  MOV     #.,HRQ.01
005067  104200  060014  001103                                                  MOV     #ERRMC,HRQ.RQ
005072  104651  000062                          MOV     U.RVER(R5),R1     GET ORIGINAL ERROR
005074  106201  000003                          CMP     #3,R1           ; SEE IF REVECTOR
005076  055103                                  BNE     7$              ; IF NOT, BRANCH
005077                                          CERROR  5,#SER26        ; REPORT REVECTOR
005077  104200  003040  001110                                                  MOV     #SER26,HRQ.05
005102  005106                                  BR      8$              ; BRANCH
005103                                  7$:     CERROR  5,#SER32        ; REPORT HEADER COMPARE ERROR
005103  104200  003054  001110                                                  MOV     #SER32,HRQ.05
005106                                  8$:     ERRORC  <CURBN,CURBN+1,U.CBN(R5),U.CBN+1(R5)>
005106  104300  002175  001111                                                  MOV     CURBN,HRQ.06
005111  104300  002176  001112                                                  MOV     CURBN+1,HRQ.07
005114  104650  000053  001113                                                  MOV     U.CBN(R5),HRQ.08
005117  104650  000054  001114                                                  MOV     U.CBN+1(R5),HRQ.09
005122  005133                                  BR      9$              ; REPORT ERROR
005123  114002                          6$:     CLR     R2              ; NO ERRORS
005124  024203                                  CALL    ENABLE          ; ENABLE ERROR RECOVERY
```

NXTRCT - GET THE NEXT RCT SECTOR TO SEARCH

```
     45 005125  104203  177777                 MOV     #-1,R3          ; START READ ATTEMPT RETRIES AT 0
     46 005127  100653  000012                 MOV     R3,U.RWTO(R5)   ; SAVE
     47 005131  104207  000023                 MOV     #SEEK,R0        ; NEXT MODULE IS SEEK
     48 005133  000000                 9$:     RETURN
     49                                        .IF     LE,BUFARA-.
     50                                 BUFARA =       .+1
     51                                        .ENDC
     52                                        .IF     LE,MAXADR-.
     53                                 MAXADR =       .+1
     54                                        .ENDC
     64
```

```
      1                                  .SBTTL  RVFAIL - CLEAR ALL REVECTOR BITS, AND CALL SETUP NEXT
      2 005134                           RVFAIL:
      3                                  ;
      4                                  ;       CLEAR ALL REVECTOR BITS, AND RESTORE U.CBN TO WHAT IT WAS
      5                                  ;
      6 005134  104651  000046                   MOV     U.PARM(R5),R1   ; GET UNIT PARAMETERS
      7 005136  103201  000400                   BIC     #REVEC,R1       ; CLEAR ALL REVECTOR BITS
      8 005140  100651  000046                   MOV     R1,U.PARM(R5)   ; SAVE
      9 005142  104201  000001                   MOV     #1,R1           ; FLAG THAT A SECTOR HAS BEEN READ (SKIPPED)
     10 005144  100651  000021                   MOV     R1,U.NSEC(R5)   ; SAVE
     11 005146  104651  000024                   MOV     U.CSEC(R5),R1   ; GET NUMBER OF SECTORS
     12 005150  105651  000051                   ADD     U.MBN(R5),R1    ; ADD STARTING SECTOR NUMBER
     13 005152  100651  000053                   MOV     R1,U.CBN(R5)    ; RESTORE CBN
     14 005154  104651  000052                   MOV     U.MBN+1(R5),R1  ; GET HI MBN
     15 005156  045160                           BCC     1$              ; IF NO CARRY, BRANCH
     16 005157  115401                           INC     R1              ; PROPOGATE CARRY
     17 005160  100651  000054          1$:      MOV     R1,U.CBN+1(R5)  ; SAVE HI IN CBN
     18 005162  104207  005522                   MOV     #L$,R0          ; GET LBN STRING
     19 005164  100647  000005                   MOV     R0,S.LETR(R4)   ; SAVE
     20 005166  104207  000001                   MOV     #SETUP,R0       ; SETUP NEXT ROUTINE
     21 005170  000000                           RETURN                  ; RETURN TO CALLING PROGRAM
```

********** OVERLAY MODULE SEEK - IF NECESSARY, ISSUE SEEK

```
                                        .SBTTL  ********** OVERLAY MODULE SEEK - IF NECESSARY, ISSUE SEEK
                                                DMOVLY  SK,BUFARA
005171  000105                                  .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;
                                        ;
                                        ;
000023                                  SEEK    =       REVCT+1         ; SEEK OVERLAY
                                        ;
                                        ;       SEEK ISSUES A SEEK TO THE CYLINDER SPECIFIED IN U.CCYL(R5) (2 WORDS)
                                        ;       AND GROUP GIVEN IN U.CGRP(R5) (1 WORD).
                                        ;       IT THEN RETURNS WITH A DEFERED CALL TO SEKTST (TEST SEEK) TO SEE IF
                                        ;       THE SEEK COMPLETED SUCCESSFULLY.  IF SO, THE READ WRITE MODULE IS
                                        ;       THEN CALLED
                                        ;
                                                .ENABL  LSB
005237  104653  000046                        MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
005241  102203  002000                        BIT     #SEKINP,R3      ; SEE IF SEEK IS ALLREADY IN PROGRESS
005243  055250                                BNE     3$              ; IF SO, BRANCH
005244  104201  177777                        MOV     #-1,R1          ; FOR ZEROING FOLLOWING WORDS
005246  100651  000007                        MOV     R1,U.SRTY(R5)   ; SAVE
005250  104653  000047                3$:     MOV     U.RCOV(R5),R3   ; GET RECOVERY WORD
005252  102203  002000                        BIT     #RCBREQ,R3      ; SEE IF RECALIBRATION REQUESTED
005254  015261                                BEQ     4$              ; IF NOT, BRANCH
005255  104207  000025                        MOV     #RECALB,R0      ; RECALIBRATE NEXT MODULE
005257  114002                                CLR     R2              ; NO ERRORS
005260  005431                                BR      SEKOUT          ; EXIT
005261  104652  000007                4$:     MOV     U.SRTY(R5),R2   ; GET SEEK RETRIES
005263  115402                                INC     R2              ; ADJUST FOR TEST
005264  015400                                BEQ     5$              ; IF FIRST TIME, BRANCH
005265  106202  000002                        CMP     #2,R2           ; SEE IF TRIED MAX NUM OF TIMES
005267  045356                                BCC     1$              ; IF NOT, BRANCH
005270                                        DEVFTL  14,<S.LETR(R4),U.CBN(R5),U.CBN+1(R5)>
005270  104200  001440  001107                                        MOV     #ER14,HRQ.04
005273  104640  000005  001110                                        MOV     S.LETR(R4),HRQ.05
005276  104650  000053  001111                                        MOV     U.CBN(R5),HRQ.06
005301  104650  000054  001112                                        MOV     U.CBN+1(R5),HRQ.07
005304  104202  047656                                              MOV     #14!FTLDEV+4000.,R2
005306  104020  001105                                              MOV     R2,HRQ.02
005310  104200  005310  001104                                      MOV     #.,HRQ.01
005313  104200  060014  001103                                      MOV     #ERRMC,HRQ.RQ
005316                                        ERRORC  <U.PARM(R5),#RBNTXT,U.RBN(R5),U.RBN+1(R5),GROUP,CYL,CYL+1>
005316  104650  000046  001113                                      MOV     U.PARM(R5),HRQ.08
005321  104200  005475  001114                                      MOV     #RBNTXT,HRQ.09
005324  104650  000055  001115                                      MOV     U.RBN(R5),HRQ.10
005327  104650  000056  001116                                      MOV     U.RBN+1(R5),HRQ.11
005332  104300  002203  001117                                      MOV     GROUP,HRQ.12
005335  104300  002201  001120                                      MOV     CYL,HRQ.13
005340  104300  002202  001121                                      MOV     CYL+1,HRQ.14
005343                                        ENDERR  0
005343  114000  002230                                              CLR     ERRPOS          ; CLEAR THE POSITION
005345  104653  000046                        MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
005347  103203  002000                        BIC     #SEKINP,R3      ; CLEAR SEEK IN PROGRESS BIT
005351  100653  000046                        MOV     R3,U.PARM(R5)   ; SAVE
```

********** OVERLAY MODULE SEEK - IF NECESSARY, ISSUE SEEK

```
 48 005353  024235                        CALL    GORCLB          ; RECALIBRATE DRIVE
 49 005354  104051                        MOV     R5,R1           ; DEFFERED CALL
 50 005355  005432                        BR      SEKEXT          ; EXIT
 51 005356                          1$:   REPSFT  ,,SEEK          ; REPORT SEEK ERROR
    005356  114000  001105                                              CLR     HRQ.02
    005360  114000  001106                                              CLR     HRQ.03
    005362  104200  000001  001107                                      MOV     #1,HRQ.04
    005365  100467                                                              MOV R0,-(SP)
    005366  104657  000063                                              MOV     U.UNUM(R5),R0
    005370  105657  000050                                              ADD     U.SUBU(R5),R0
    005372  104070  001104                                              MOV     R0,HRQ.01
    005374  104207  060007                                              MOV     #T4SOFT,R0
    005376  021053                                                      CALL    HOSTRQ
    005377  104267                                                              MOV (SP)+,R0
 52 005400  025433                    5$:   CALL    TSTNEC          ; SEE IF SEEK IS NECESSARY
 53 005401  115003                          TST     R3              ; SEE IF SEEK IS NECESSARY
 54 005402  015426                          BEQ     NOSEEK          ; IF ZERO (NO SEEK NECESSARY), BRANCH
 55 005403  025502                          CALL    ISUSEK          ; ISSUE SEEK
 56 005404  115002                          TST     R2              ; SEE IF ERROR OCCURRED
 57 005405  055432                          BNE     SEKEXT          ; IF ERROR, BRANCH
 58 005406  104651  000007                  MOV     U.SRTY(R5),R1   ; GET SEEK TIMEOUT
 59 005410  115401                          INC     R1              ; INCREMENT COUNT
 60 005411  100651  000007                  MOV     R1,U.SRTY(R5)   ; SAVE
 61 005413  104657  000011                  MOV     U.MSTO(R5),R0   ; MOVE TIMEOUT TO PARAMETERS
 62 005415  100657  000005                  MOV     R0,U.TIMH(R5)   ; INITILIZE TIMEOUT VALUE
 63                                 ;      NOTE:   R2 IS ZERO HERE
 64 005417  100652  000006                  MOV     R2,U.TIML(R5)   ; SAVE
 65 005421  104051                          MOV     R5,R1           ; DEFERRED CALL TO SEKTST
 66 005422  104207  000024                  MOV     #SEKTST,R0      ; SEKTST NEXT TO CALL
 67 005424  024212                          CALL    DSABLE          ; DISABLE ERROR RECOVERY DURING SEEK
 68 005425  005432                          BR      SEKEXT          ; BRANCH
 69 005426  104207  000007          NOSEEK: MOV     #BUILDP,R0      ; READ/WRITE ROUTINE NEXT
 70 005430  114002                          CLR     R2              ; NO ERRORS
 71 005431  114001                  SEKOUT: CLR     R1              ; IMMIDATE CALL
 72 005432  003231                  SEKEXT: BR      JMPRET          ; RETURN TO CALLING PROGRAM
 73                                         .DSABL  LSB
```

```
                                        .SBTTL  TSTNEC - TEST TO SEE IF SEEK IS NECESSARY
005433                                  TSTNEC:
                                        ;
                                        ;       TSTNEC TESTS TO SEE IF A SEEK IS NECESSARY.  IF NOT, R3 IS RETURNED
                                        ;       AS ZERO, NONZERO OTHERWISE.
                                        ;
005433  104657  000046                        MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
005435  102207  000200                        BIT     #RBNBN,R0       ; SEE IF BLOCK REVECTORED
005437  015447                              BEQ     2$              ; IF NOT, BRANCH
005440  104650  000055  002175                MOV     U.RBN(R5),CURBN ; MOVE RBN TO CALCULATION AREA
005443  104650  000056  002176                MOV     U.RBN+1(R5),CURBN+1     ; MOVE RBN TO CALCULATION AREA
005446  005455                              BR      3$              ; BRANCH
005447  104650  000053  002175  2$:           MOV     U.CBN(R5),CURBN ; MOVE LBN TO CALCULATION AREA
005452  104650  000054  002176                MOV     U.CBN+1(R5),CURBN+1     ; MOVE LBN TO CALCULATION AREA
005455  114007                        3$:     CLR     R0              ; TELL CALC ROUTINE TO SET UP PARAMETERS
005456  022027                              CALL    CALC            ; CALCULATE CYL AND GROUP
005457  104203  002201                        MOV     #CYL,R3         ; R3 POINTS TO CALCULATED CYLINDER
005461  104657  000047                        MOV     U.RCOV(R5),R0   ; GET UNIT PARAMETERS
005463  102207  004000                        BIT     #SEKREQ,R0      ; SEE IF SEEK MANDATORY
005465  055501                              BNE     9$              ; IF SO, BRANCH
005466  104202  000064                        MOV     #U.CCYL,R2      ; R2 WILL POINT TO CURRENT CYL
005470  105052                              ADD     R5,R2           ; R2 POINTS TO CURRENT CYLINDER
005471  104201  000003                        MOV     #3,R1           ; GET COUNT
005473  104237                        4$:     MOV     (R3)+,R0        ; GET WORD
005474  106227                              CMP     (R2)+,R0        ; SEE IF CYL AND GROUP THE SAME
005475  055501                              BNE     9$              ; IF NOT, BRANCH
005476  117401                              DEC     R1              ; DECREMENT COUNT
005477  055473                              BNE     4$              ; IF COUNT INCOMPLETE, BRANCH
005500  114003                              CLR     R3              ; FLAG AS SEEK NOT NECESSARY
005501  000000                  9$:         RETURN                  ; RETURN TO SEEK
```

```
                                        .SBTTL  ISUSEK - ISSUE SEEK COMMAND
005502                                  ISUSEK:
                                        ;
                                        ;       ISUSEK ISSUES THE SEEK TO THE DRIVE
                                        ;
005502  104300  002201  001673                 MOV     CYL,INS+1       ; MOVE LOW CYL TO SEEK COMMAND
005505  104300  002202  001674                 MOV     CYL+1,INS+2     ; SET LOWER BITS
005510  104300  002203  001675                 MOV     GROUP,INS+3     ; MOVE GROUP TO SEEK COMMAND
005513  104203  001636                         MOV     #CR.SEK,R3      ; SET UP FOR TALK
005515  021173                                 CALL    TALK            ; SEND SEEK
005516  115002                                 TST     R2              ; SEE IF ERROR OCCURRED
005517  055555                                 BNE     10$             ; IF SO, BRANCH
005520  104207  000064                         MOV     #U.CCYL,R0      ; R0 WILL POINT TO CURRENT CYLINDER
005522  105057                                 ADD     R5,R0           ; R0 POINTS TO CURRENT CYLINDER
005523  104201  000003                         MOV     #3,R1           ; MOVE THREE WORDS
005525                                         PUSH    <R0,R1>         ; SAVE POINTER AND COUNT
005525  100467                                                                 MOV R0,-(SP)
005526  100461                                                                 MOV R1,-(SP)
005527  104202  000067                         MOV     #U.LCYL,R2      ; R2 WILL POINT TO LAST CYLINDER
005531  105052                                 ADD     R5,R2           ; R2 POINTS TO LAST CYLINDER
005532  104273                         1$:     MOV     (R0)+,R3        ; GET WORD
005533  100223                                 MOV     R3,(R2)+        ; SAVE
005534  117401                                 DEC     R1              ; DECREMENT COUNT
005535  055532                                 BNE     1$              ; IF COUNT INCOMPLETE, BRANCH
005536                                         POP     <R2,R0>         ; RESTORE
005536  104262                                                                 MOV (SP)+,R2
005537  104267                                                                 MOV (SP)+,R0
005540  104201  002201                 4$:     MOV     #CYL,R1         ; R2 POINTS TO NEW CYL
005542  104213                         2$:     MOV     (R1)+,R3        ; GET WORD
005543  100273                                 MOV     R3,(R0)+        ; SAVE
005544  117402                                 DEC     R2              ; DECREMENT COUNT
005545  055542                                 BNE     2$              ; IF COUNT INCOMPLETE, BRANCH
005546  104653  000046                         MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
005550  101203  002000                         BIS     #SEKINP,R3      ; MARK SEEK IN PROGRESS
005552  100653  000046                         MOV     R3,U.PARM(R5)   ; SAVE
005554  005560                                 BR      ISUEXT          ; BRANCH (NOTE THAT R2 IS ZERO - NO ERRORS)
005555                                 10$:    CERROR  5,#SER9         ; REPORT SECONDARY ERROR
005555  104200  005053  001110                                                 MOV     #SER9,HRQ.05
005560  000000                         ISUEXT: RETURN                  ; RETURN TO CALLING PROGRAM
                                               .IF     LE,MAXADR-.
                                       MAXADR  =       .+1
                                               .ENDC
```

********** OVERLAY MODULE SEKTST - SEE IF THE SEEK IS COMPLETE

```
                                .SBTTL  ********** OVERLAY MODULE SEKTST - SEE IF THE SEEK IS COMPLETE
                                        DMOVLY  TS,BUFARA
005561  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
                                ;
000024                          SEKTST  =       SEEK+1          ; SEEK COMPLETE TEST OVERLAY
                                ;
                                ;       SEKTST TESTS TO SEE IF READ/WRITE READY OR ATTENTION IS HIGH
                                ;       R2 RETURNED AS ZERO IF SEEK WAS SUCESSFUL, -1 IF SEEK INCOMPLETE
                                ;       POSITIVE NON-ZERO IF AN ERROR WAS DETECTED
                                ;
                                        .ENABL  LSB
005237  020746                         CALL    RTDS            ; GET REAL TIME  DRIVE STATE
005240  115002                         TST     R2              ; SEE IF ERROR OCCURRED
005241  015246                         BEQ     1$              ; IF NOT, BRANCH
005242  104207  000024                 MOV     #SEKTST,R0      ; RETRY THIS MODULE
005244  104051                         MOV     R5,R1           ; DEFERRED CALL
005245  005626                         BR      STSEXT          ; EXIT
005246  024203                 1$:     CALL    ENABLE          ; ENABLE ERROR RECOVERY
005247  115001                         TST     R1              ; SEE IF R/W RDY ASSERTED
005250                                 ASSUME  RWRDY,100000    ; ASSUME R/W RDY IS SIGN BIT
005250  035426                         BPL     STSERR          ; IF NOT READY, BRANCH
005251  104657  000046                 MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
005253  103207  002000                 BIC     #SEKINP,R0      ; SEEK NO LONGER IN PROGRESS
005255  100657  000046                 MOV     R0,U.PARM(R5)   ; SAVE PARAMETERS
005257  104657  000047                 MOV     U.RCOV(R5),R0   ; GET RECOVERY WORD
005261  103207  004000                 BIC     #SEKREQ,R0      ; SEEK NO LONGER REQUIRED
005263  100657  000047                 MOV     R0,U.RCOV(R5)   ; SAVE
005265  104647  000001                 MOV     S.SEEK(R4),R0   ; GET NUMBER OF SEEKS ISSUED
005267  117407                         DEC     R0              ; DECREMENT SEEK COUNT
005270  055304                         BNE     SEKCNT          ; IF NO REPORT NEEDED, BRANCH
005271  104650  000063  001104         MOV     U.UNUM(R5),HRQ.01 ; GET STAPTING SUBUNIT NUMBER
005274  105650  000050  001104         ADD     U.SUBU(R5),HRQ.01 ; ADD OFFSET
005277  104207  060010                 MOV     #T4SEEK,R0      ; MOVE SEEK REPORT NUMBER TO R0
005301  021053                         CALL    HOSTRQ          ; REPORT TO HOST
005302  104207  001750                 MOV     #1000.,R0       ; SET UP FOR NEW COUNT
                                        .DSABL  LSB
005304  100647  000001         SEKCNT: MOV     R0,S.SEEK(R4)   ; SAVE COUNT
005306  114002                         CLR     R2              ; NO ERRORS
005307  104653  000007                 MOV     U.SRTY(R5),R3   ; GET RETRY COUNT
005311  015413                         BEQ     2$              ; IF NO RETRIES, BRANCH
005312                                 REPSFT  SOFT            ; REPORT SOFT ERROR
005312  104200  000001  001105                                         MOV     #1,HRQ.02
005315  114000  001106                                                 CLR     HRQ.03
005317  114000  001107                                                 CLR     HRQ.04
005321  100467                                                                 MOV R0,-(SP)
005322  104657  000063                                                 MOV     U.UNUM(R5),R0
005324  105657  000050                                                 ADD     U.SUBU(R5),R0
005326  104070  001104                                                 MOV     R0,HRQ.01
005330  104207  060007                                                 MOV     #T4SOFT,R0
005332  021053                                                         CALL    HOSTRQ
005333  104267                                                                 MOV (SP)+,R0
```

```
     54 005334                                          SOFTER  15,<R3,S.LETR(R4),U.CBN(R5),U.CBN+1(R5)>
        005334  104200  001506  001107                                                MOV     #ER15,HRQ.04
        005337  104030  001110                                                        MOV     R3,HRQ.05
        005341  104640  000005  001111                                                MOV     S.LETR(R4),HRQ.06
        005344  104650  000053  001112                                                MOV     U.CBN(R5),HRQ.07
        005347  104650  000054  001113                                                MOV     U.CBN+1(R5),HRQ.08
        005352  104202  147657                                                        MOV     #15!ERSOFT+4000.,R2
        005354  104020  001105                                                        MOV     R2,HRQ.02
        005356  104200  005356  001104                                                MOV     #.,HRQ.01
        005361  104200  060013  001103                                                MOV     #ERRMES,HRQ.RQ
     55 005364                                          ERRORC  <U.PARM(R5),#RBNTXT,U.RBN(R5),U.RBN+1(R5),U.CGRP(R5),U.CCYL(R5)>
        005364  104650  000046  001114                                                MOV     U.PARM(R5),HRQ.09
        005367  104200  005475  001115                                                MOV     #RBNTXT,HRQ.10
        005372  104650  000055  001116                                                MOV     U.RBN(R5),HRQ.11
        005375  104650  000056  001117                                                MOV     U.RBN+1(R5),HRQ.12
        005400  104650  000066  001120                                                MOV     U.CGRP(R5),HRQ.13
        005403  104650  000064  001121                                                MOV     U.CCYL(R5),HRQ.14
     56 005406                                          ERRORC  U.CCYL+1(R5)
        005406  104650  000065  001122                                                MOV     U.CCYL+1(R5),HRQ.15
     57 005411                                          ENDERR  0
        005411  114000  002230                                                        CLR     ERRPOS          ; CLEAR THE POSITION
     58 005413  104207  000007          2$:     MOV     #BUILDP,R0      ; BUILD THE READ/WRITE CHAIN NEXT
     59 005415  104653  000047                  MOV     U.RCOV(R5),R3   ; GET RECOVERY WORDS
     60 005417  102203  030000                  BIT     #LEVUSD!NXTLEV,R3       ; SEE IF ERROR RECOVERY LEVELS IN USE
     61 005421  015424                          BEQ     1$              ; IF NOT, BRANCH
     62 005422  104207  000017                  MOV     #NEWLEV,R0      ; ISSUE ERROR RECOVERY COMMAND NEXT
     63 005424  114001          1$:     CLR     R1              ; IMMIDATE CALL TO NEXT MODULE
     64 005425  005626                          BR      STSEXT          ; BRANCH
     65 005426  102201  000100          STSERR: BIT     #AVAIL,R1       ; SEE IF DRIVE TIMEOUT HAS EXPIRED
     66 005430  015442                          BEQ     1$              ; IF NOT, BRANCH
     67 005431  104653  000046                  MOV     U.PARM(R5),R3   ; GET UNIT PARAMETERS
     68 005433  103203  002000                  BIC     #SEKINP,R3      ; MARK AS SEEK NOT IN PROGRESS
     69 005435  100653  000046                  MOV     R3,U.PARM(R5)   ; SAVE
     70 005437  104207  000023                  MOV     #SEEK,R0        ; SEEK AGAIN
     71 005441  005626                          BR      STSEXT          ; EXIT
     72 005442  102201  000002          1$:     BIT     #ATTN,R1        ; SEE IF ATTENTION ASSERTED
     73 005444  015536                          BEQ     2$              ; IF NOT, BRANCH
     74 005445                                  REPSFT  SOFT,,SEEK      ; REPORT SEEK AND SOFT ERROR
        005445  104200  000001  001105                                                MOV     #1,HRQ.02
        005450  114000  001106                                                        CLR     HRQ.03
        005452  104200  000001  001107                                                MOV     #1,HRQ.04
        005455  100467                                                                          MOV R0,-(SP)
        005456  104657  000063                                                        MOV     U.UNUM(R5),R0
        005460  105657  000050                                                        ADD     U.SUBU(R5),R0
        005462  104070  001104                                                        MOV     R0,HRQ.01
        005464  104207  060007                                                        MOV     #T4SOFT,R0
        005466  021053                                                                CALL    HOSTRQ
        005467  104267                                                                          MOV (SP)+,R0
     75 005470                                          SOFTER  1,<U.LGRP(R5),U.LCYL(R5),U.LCYL+1(R5),U.CGRP(R5),U.CCYL(R5)>
        005470  104200  000000  001107                                                MOV     #ER1,HRQ.04
        005473  104650  000071  001110                                                MOV     U.LGRP(R5),HRQ.05
        005476  104650  000067  001111                                                MOV     U.LCYL(R5),HRQ.06
        005501  104650  000070  001112                                                MOV     U.LCYL+1(R5),HRQ.07
        005504  104650  000066  001113                                                MOV     U.CGRP(R5),HRQ.08
        005507  104650  000064  001114                                                MOV     U.CCYL(R5),HRQ.09
        005512  104202  147641                                                        MOV     #1!ERSOFT+4000.,R2
        005514  104020  001105                                                        MOV     R2,HRQ.02
```

```
                005516  104200  005516  001104                                                  MOV     #.,HRQ.01
                005521  104200  060013  001103                                                  MOV     #ERRMES,HRQ.RQ
     76 005524                                  ERRORC  U.CCYL+1(R5)
                005524  104650  000065  001115                                                  MOV     U.CCYL+1(R5),HRQ.10
     77 005527                                  ENDERR                  ; REPORT ATTN ASSERTED DURING SEEK
                005527  104200  000051  001116                                                  MOV     #SER22,HRQ.11
                005532  104200  000014  002230                                                  MOV     #11+1,ERRPOS    ; SET THE POSITION
     78 005535  005626                          BR      STSEXT          ; EXIT
     79 005536  024212                  2$:     CALL    DSABLE
     80 005537  104051                          MOV     R5,R1           ; MAKE R1 NON-ZERO FOR DEFERRED CALL
     81                                 ;        NOTE:   R2 IS ZERO AT THIS POINT
     82 005540  104207  000024                  MOV     #SEKTST,R0      ; SEEK TEST IS NEXT MODULE
     83 005542  104653  000006                  MOV     U.TIML(R5),R3   ; GET TIMEOUT VALUE
     84 005544  107203  000001                  SUB     #1,R3           ; DECREMENT TIMEOUT VALUE
     85 005546  100653  000006                  MOV     R3,U.TIML(R5)   ; SAVE TIMEOUT VALUE
     86 005550  045626                          BCC     STSEXT          ; IF NON-ZERO BRANCH
     87 005551  104653  000005                  MOV     U.TIMH(R5),R3   ; GET HI ORDER TIMEOUT
     88 005553  107203  000001                  SUB     #1,R3           ; DECREMENT COUNT
     89 005555  100653  000005                  MOV     R3,U.TIMH(R5)   ; SAVE
     90 005557  045626                          BCC     STSEXT
     91 005560                                  SOFTER  3,<U.LGRP(R5),U.LCYL(R5),U.LCYL+1(R5)>
                005560  104200  000166  001107                                                  MOV     #ER3,HRQ.04
                005563  104650  000071  001110                                                  MOV     U.LGRP(R5),HRQ.05
                005566  104650  000067  001111                                                  MOV     U.LCYL(R5),HRQ.06
                005571  104650  000070  001112                                                  MOV     U.LCYL+1(R5),HRQ.07
                005574  104202  147643                                                          MOV     #3!ERSOFT+4000.,R2
                005576  104020  001105                                                          MOV     R2,HRQ.02
                005600  104200  005600  001104                                                  MOV     #.,HRQ.01
                005603  104200  060013  001103                                                  MOV     #ERRMES,HRQ.RQ
     92 005606                                  ERRORC  <U.CGRP(R5),U.CCYL(R5),U.CCYL+1(R5)>
                005606  104650  000066  001113                                                  MOV     U.CGRP(R5),HRQ.08
                005611  104650  000064  001114                                                  MOV     U.CCYL(R5),HRQ.09
                005614  104650  000065  001115                                                  MOV     U.CCYL+1(R5),HRQ.10
     93 005617                                  ENDERR
                005617  104200  000051  001116                                                  MOV     #SER22,HRQ.11
                005622  104200  000014  002230                                                  MOV     #11+1,ERRPOS    ; SET THE POSITION
     94 005625  024227                          CALL    GOSEEK
     95 005626  003231                  STSEXT: BR      JMPRET          ; RETURN TO         R
     96                                         .IF     LE,MAXADR-.
     97                                 MAXADR  =       .+1
     98                                         .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE RECALB - RECALIBRATION
                                        DMOVLY  RC,BUFARA
005627  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
                                ;
000025                          RECALB  =       SEKTST+1                ; RECALIBRATION MODULE
                                ;
                                ;
                                ;
                                        .ENABL  LSB
005237  104203  001655                  MOV     #CR.INR,R3      ; POINT TO THE INITIATE RECALIBRATION COMMAND
005241  021173                          CALL    TALK            ; SEND-RECEIVE SDI COMMAND
005242  115002                          TST     R2              ; SEE IF ANY ERRORS OCCURRED
005243  015273                          BEQ     1$              ; IF NOT, BRANCH
                                ;
                                ;       FROM HERE TO THE LABEL '5$' IS CODE THAT SEES IF THE HOST WAS
                                ;       HALTED LONG ENOUGH FOR THE DRIVE TO GO OFFLINE DURING THE RECEIVE
                                ;       LEVEL 2 OPERATION (NOTE THAT LONG TIMEOUTS TALK TO THE HOST <<DURING>>
                                ;       THE TIMEOUT
                                ;
005244  106202  147747                  CMP     #71.!ERSOFT+4000.,R2    ; SEE IF TIMEOUT OF RECEIVE
005246  055267                          BNE     5$              ; IF NOT, BRANCH
005247  020746                          CALL    RTDS            ; GET STATE
005250  115002                          TST     R2              ; SEE IF ERROR
005251  015254                          BEQ     4$              ; IF NOT, BRANCH
005252  024221                          CALL    GORTRY          ; RETRY THIS MODULE
005253  005315                          BR      2$              ; EXIT
005254  102201  000100          4$:     BIT     #AVAIL,R1       ; SEE IF DRIVE IS AVAILABLE
005256  015266                          BEQ     6$              ; IF NOT, BRANCH
005257  104303  001661                  MOV     CR.INR+L2.EOF,R3        ; GET INR OFFSET
005261  105053                          ADD     R5,R3           ; POINT TO RECALIBRATE ERROR COUNT
005262  104131                          MOV     (R3),R1         ; GET COUNT
005263  117401                          DEC     R1              ; DECREMENT COUNT
005264  100131                          MOV     R1,(R3)         ; SAVE
005265  005303                          BR      3$              ;
005266  104052                  6$:     MOV     R5,R2           ; FLAG ERROR
005267                          5$:     CERROR  5,#SER7         ; REPORT SECONDARY ERROR
005267  104200  005013  001110                                                  MOV     #SER7,HRQ.05
005272  005315                          BR      2$              ; EXIT
005273  104653  000047          1$:     MOV     U.RCOV(R5),R3   ; GET RECOVERY PARAMETERS
005275  103203  002000                  BIC     #RCBREQ,R3      ; CLEAR RECALIBRATION REQUESTED FLAG
005277  101203  004000                  BIS     #SEKREQ,R3      ; MARK SEEK AS REQUIRED
005301  100653  000047                  MOV     R3,U.RCOV(R5)   ; SAVE
005303  114002                  3$:     CLR     R2              ; NO ERRORS
005304  100652  000064                  MOV     R2,U.CCYL(R5)   ; ZERO CYLINDER
005306  100652  000065                  MOV     R2,U.CCYL+1(R5)
005310  100652  000066                  MOV     R2,U.CGRP(R5)
005312  114001                          CLR     R1              ; IMMIDATE CALL
005313  104207  000023                  MOV     #SEEK,R0        ; SEEK IS NEXT MODULE
005315  003231                  2$:     BR      JMPRET          ; RETURN TO                     61
                                        .IF     LE,MAXADR-.
```

```
63                                      MA>ADR  =       .+1
64                                              .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE DRPALL - DROP ALL SUBUNITS ON THIS DRIVE
005316                                  DMOVLY  DA,AREA0
005316  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
000026                          DRPALL  =       RECALB+1
                                ;
                                ;       DROP ALL SUBUNITS ON THIS DRIVE AND REPORT
                                ;
                                ;       NOTE:
                                ;               THIS IS A VERY UNSTRUCTURED PIECE OF CODE -- IT HAS SEVERAL
                                ;               JUMPS DIRECTLY INTO         T SEVERAL DIFFERENT PLACES.
                                ;
                                        .ENABL  LSB
004422                                  PUSH    R4              ; SAVE R4 (JUST IN CASE)
004422  100464                                                                         MOV R4,-(SP)
004423  104200  005550  001105        MOV     #MS2,HRQ.02     ; POINT TO MESSAGE
004426  104207  001107                MOV     #HRQ.04,R0      ; POINT TO WHERE TO PUT UNITS
004430  104053                        MOV     R5,R3           ; GET POINTER TO UNIT DATABASE
004431  115403                        INC     R3              ; POINT TO SUBUNIT POINTERS
004432                                ASSUME  U.SUBP,1
004432  114001                        CLR     R1              ; CLEAR INDEX
004433  114004                        CLR     R4              ; CLEAR NUMBER OF SUBUNITS COUNT
004434  104232                1$:     MOV     (R3)+,R2        ; GET POINTER
004435  074445                        BMI     2$              ; IF NO SUBUNIT, BRANCH
004436  104652  000063                MOV     U.UNUM(R5),R2   ; GET BASE UNIT NUMBER
004440  105012                        ADD     R1,R2           ; ADD OFFSET
004441  100272                        MOV     R2,(R0)+        ; MOVE TO OUTPUT BUFFER
004442  104020  001104                MOV     R2,HRQ.01       ; MOVE TO UNIT NUMBER
004444  115404                        INC     R4              ; INC SUBUNIT COUNT
004445  115401                2$:     INC     R1              ; INC COUNT
004446  106201  000003                CMP     #3,R1           ; SEE IF EXPIRED
004450  044434                        BCC     1$              ; IF NOT, BRANCH
004451  104647  004467                MOV     SER18F-1(R4),R0 ; POINT TO DRIVE LIST
004453  104070  001106                MOV     R0,HRQ.03       ; POINT TO CORRECT ERROR MESSAGE
004455  104207  060015                MOV     #MESSAG,R0      ; MESSAGE TO R0
004457  021053                        CALL    HOSTRQ          ; REPORT
004460  104651  000046                MOV     U.PARM(R5),R1   ; GET UNIT PARAMETERS
004462  101201  100000                BIS     #DROP,R1        ; SET DROP BIT
004464  100651  000046                MOV     R1,U.PARM(R5)   ; SAVE
004466                                POP     R4
004466  104264                                                                         MOV (SP)+,R4
004467  003715                        BR      NOSUB

004470  005435                SER18F: .WORD   SER18A
004471  005431                        .WORD   SER18B
004472  005425                        .WORD   SER18C
004473  005421                        .WORD   SER18D
                                        .DSABL  LSB

                                        .IF     LE,MAXADR-.
                                MAXADR  =       .+1
```

```
     55                                                .ENDC
```

```
                                        .SBTTL  ********** OVERLAY MODULE INSET - SET UP UNIT FOR INITIALZATION
                                                DMOVLY  IN,AREAI
004474  000105                                  .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;
                                        ;
                                        ;
000027                                  INSET   =       DRPALL+1
                                        ;
                                        ;       INSET WILL SET UP EACH UNIT BEFORE IT STARTS RUNNING
                                        ;
                                                .ENABL  LSB
004710  104657  000046                          MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
004712  103207  004000                          BIC     #NEWSUB,R0      ; NO LONGER NEW SUBUNITS
004714  100657  000046                          MOV     R0,U.PARM(R5)   ; SAVE
004716  114002                                  CLR     R2              ; NO ERRORS
004717  100652  000064                          MOV     R2,U.CCYL(R5)   ; CLEAR CYL+GROUP
004721  100652  000065                          MOV     R2,U.CCYL+1(R5)
004723  100652  000066                          MOV     R2,U.CGRP(R5)
004725  114001                                  CLR     R1              ; IMMIDATE CALL
004726  104207  000030                          MOV     #COMCHR,R0      ; COMMON CHARACTERISTICS NEXT MODULE CALLED
004730  003231                                  BR      JMPRET          ; RETURN TO         R
                                                .DSABL  LSB
                                                .IF     LE,MAXADR-.
                                        MAXADR  =       .+1
                                                .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE COMCHR - SET UP COMMON CHARACTERISTICS
004731                                  DMOVLY  CC,AREAI
004731  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
                                ;
000030                          COMCHR  =       INSET+1
                                ;
                                ;       GET COMMON CHARACTERISTICS AND SET UP THE ERROR RECOVERY LEVEL,
                                ;       RETRIES, LONG TIMEOUT, AND SHORT TIMEOUT
                                ;
                                        .ENABL  LSB
004710  104203  004462                  MOV     #CR.GCR,R3      ; POINT TO GET CHARACTERISTICS DATA BLOCK
004712  021173                          CALL    TALK            ; GET CHARACTERISTICS
004713  115002                          TST     R2              ; SEE IF ANY ERRORS OCCURRED
004714  014732                          BEQ     1$              ; IF NOT, BRANCH
004715                                  CERROR  5,#SER8         ; REPORT SECONDARY ERROR
004715  104200  005032  001110                                                  MOV     #SER8,HRQ.05
004720  103200  100000  001105          BIC     #C2DFTL,HRQ.02  ; CHANGE TO HARD ERROR
004723  104200  060014  001103          MOV     #ERRMC,HRQ.RQ   ; COUNT ERROR
004726  101200  000002  002223          BIS     #DIE,M.PARM     ; FLAG INITIALIZATION ERROR
004731  005036                          BR      2$              ; BRANCH TO EXIT
004732  104300  001706  004525  1$:     MOV     ST+DRVID,DSERNM ; SAVE DRIVE SERIAL NUMBER
004735  104300  001707  004526          MOV     ST+DRVID+1,DSERNM+1     ; SAVE DRIVE SERIAL NUMBER
004740  104300  001710  004527          MOV     ST+DRVID+2,DSERNM+2     ; SAVE DRIVE SERIAL NUMBER
004743  104307  001703                  MOV     ST+RETS,R0      ; GET NUMBER OF RETRIES ALLOWED
004745  110607                          ROR     R0              ; ROTATE TO CORRECT POSITION
004746  110607                          ROR     R0
004747  110607                          ROR     R0
004750  110607                          ROR     R0
004751  103207  177760                  BIC     #LBLONB,R0      ; CLEAR UNUSED BITS
004753  100657  000030                  MOV     R0,U.RTRY(R5)   ; SAVE IN UNIT PARAMETERS
004755  104307  001704                  MOV     ST+ERLEV,R0     ; GET ERROR RECOVERY LEVELS
004757  103207  177400                  BIC     #HIBYTE,R0      ; CLEAR UNUSED BITS
004761  100657  000031                  MOV     R0,U.MLEV(R5)   ; SAVE IN UNIT PARAMETERS
004763  104307  001704                  MOV     ST+ECCRSH,R0    ; GET ECC THRESHOLD
004765  110707                          SWAB    R0              ; MOVE ECC THRESHOLD TO LOWER BYTE
004766  103207  177400                  BIC     #HIBYTE,R0      ; CLEAR UNUSED BITS
004770  100657  000032                  MOV     R0,U.ECCT(R5)   ; SAVE
004772  104307  001703                  MOV     ST+LONGTO,R0    ; GET LONG TIMEOUT
004774  103207  177760                  BIC     #LBLONB,R0      ; CLEAR UNUSED BITS
004776  025040                          CALL    TO              ; CALCULATE TIMEOUT
004777  100657  000034                  MOV     R0,U.SDIL(R5)   ; SAVE IN UNIT PARAMETERS
005001  104307  001702                  MOV     ST+SHRTTO,R0    ; GET SHORT TIMEOUT
005003  103207  177760                  BIC     #LBLONB,R0      ; CLEAR UNUSED BITS
005005  025040                          CALL    TO              ; CALCULATE TIMEOUT
005006  100657  000033                  MOV     R0,U.SDIS(R5)   ; SAVE SHORT TIMEOUT IN UNIT PARAMETERS
005010  114001                          CLR     R1              ; CLEAR LO SEEK TIMEOUT
005011  114002                          CLR     R2              ; CLEAR HI TIMEOUT
005012  105201  025762          4$:     ADD     #11250.,R1      ; ADD 9 SEC TIMEOUT TO LO ORDER TIMEOUT
005014  045016                          BCC     3$              ; IF NO CARRY, BRANCH
005015  115402                          INC     R2              ; PROPOGATE CARRY
005016  117407                  3$:     DEC     R0              ; DECREMENT TIMEOUT
```

```
     56 005017  055012                          BNE     4$              ; IF UNEXPIRED  BRANCH
     57 005020  115402                          INC     R2              ; ROUND UP
     58 005021  100652  000011                  MOV     R2,U.MSTO(R5)   ; SAVE SEEK MASTER TIMEOUT
     59 005023  104307  001703                  MOV     ST+RCTCPS,R0    ; GET NUMBER OF RCT COPIES
     60 005025  110707                          SWAB    R0              ; MOVE TO LOW WORD
     61 005026  103207  177760                  BIC     #LBLONB,R0      ; CLEAR UNUSED BITS
     62 005030  117407                          DEC     R0              ; ADJUST FOR TEST 4 INTERNALS
     63 005031  100657  000057                  MOV     R0,U.COPY(R5)   ; SAVE
     64 005033  104207  C00031                  MOV     #SPINUP,R0      ; SPINUP NEXT MODULE
     65 005035  114002                          CLR     R2              ; NO ERRORS
     66 005036  114001                  2$:     CLR     R1              ; IMMIDATE CALL TO NEXT MODULE
     67 005037  003231                          BR      JMPRET          ; RETURN TO
     68                                         .DSABL  LSB
     69 005040                          TO:
     70                                 ;
     71                                 ;       CALCULATE THE TIMEOUT IN 9SEC INTERVALS (UDA SDI RECEIVE XFC TAKES
     72                                 ;       9 SEC)
     73                                 ;
     74 005040  104201  000001                  MOV     #1,R1           ; SET UP LOG2 SHIFTER
     75 005042  105011                  1$:     ADD     R1,R1           ; DOUBLE THE TIMEOUT VALUE
     76 005043  117407                          DEC     R0              ; DECREMENT COUNT
     77 005044  055042                          BNE     1$              ; IF COUNT INCOMPLETE, BRANCH
     78 005045  115407                  2$:     INC     R0              ; INCREMENT 9 SEC COUNT
     79 005046  107201  000011                  SUB     #9.,R1          ; SUBTRACT 9 SEC FROM TIMEOUT
     80 005050  035045                          BPL     2$              ; IF MORE TIME TO GO, BRANCH
     81 005051  000000                          RETURN                  ; RETURN TO CALLING PROGRAM
     82                                         .IF     LE,MAXADR-.
     83                                 MAXADR  =       .+1
     84                                         .ENDC
```

```
                                        .SBTTL  ********** OVERLAY MODULE SPINUP - SPIN THE DRIVE UP
005052                                          DMOVLY  SP,AREAI
005052  000105                                  .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;
                                        ;
                                        ;
000031                                  SPINUP  =       COMCHR+1
                                        ;
                                        ;       SPINUP WILL SPIN THE DRIV UP IF SPUN DOWN
                                        ;
                                                .ENABL  LSB
004710  104203  004474                          MOV     #CR.RUN,R3      ; POINT TO RUN COMMAND
004712  021173                                  CALL    TALK            ; SEND COMMAND
004713  115002                                  TST     R2              ; SEE IF ERROR OCCURRED
004714  054720                                  BNE     1$              ; IF SO, BRANCH
004715  104207  000032                          MOV     #SORT,R0        ; SORT NEXT MODULE
004717  004734                                  BR      2$              ; EXIT
004720                                  1$:     CERROR  5,#SER6         ; REPORT SECONDARY ERROR
004720  104200  004773  001110                                                  MOV     #SER6,HRQ.05
004723  103200  100000  001105                  BIC     #C2DFTL,HRQ.02  ; CHANGE TO HARD ERROR
004726  104200  060014  001103                  MOV     #ERRMC,HRQ.RQ   ; COUNT ERROR
004731  101200  000002  002223                  BIS     #DIE,M.PARM     ; FLAG INITIALIZATION ERROR
004734  114001                          2$:     CLF     R1              ; IMMIDATE CALL
004735  003231                                  BR      JMPRET          ; RETURN TO                 27
                                                .IF     LE,MAXADR-.
                                        MAXADR  =       .+1
                                                .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE SORT - SORT ALL CYLINDERS, BEGIN/SETS, AND BAD BLOCKS
004736                                  DMOVLY  SO,AREAI
004736  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
                                ;
000032                          SORT    =       SPINUP+1
                                ;
                                ;       SORT WILL SORT ALL BEGIN/END SETS, BAD BLOCKS AND TRACKS/GROUPS IN
                                ;       ASCENDING ORDER
                                ;
                                        .ENABL  LSB
004710                                  PUSH    <R4,R5>         ; SAVE R4 AND R5
004710  100464                                                                  MOV R4,-(SP)
004711  100465                                                                  MOV R5,-(SP)
004712  115405                          INC     R5              ; POINT TO SUBUNIT POINTERS
004713                                  ASSUME  U.SUBP,1
004713  104204  000004                 MOV     #4,R4           ; MAXIMUM OF 4 SUBUNITS
004715  104257                  4$:     MOV     (R5)+,R0        ; R0 POINTS TO SUBUNIT DATABASE
004716  074733                          BMI     3$              ; IF NO SUBUNIT, BRANCH
004717                                  PUSH    <R4,R5>         ; SAVE R4 AND R5
004717  100464                                                                  MOV R4,-(SP)
004720  100465                                                                  MOV R5,-(SP)
004721  104171                          MOV     (R0),R1         ; GET SUBUNIT PARAMETERS
004722                                  ASSUME  S.PARM,0
004722  102201  000040                 BIT     #BEUSED,R1      ; SEE IF BEGIN/END SETS USED
004724  054727                          BNE     1$              ; IF SO, BRANCH
004725  025103                          CALL    SORTTG          ; SORT THE TRACKS/GROUPS
004726  004730                          BR      2$              ; BRANCH
004727  024744                  1$:     CALL    SORTBE          ; SORT THE BEGIN/END SETS
004730  025034                  2$:     CALL    SORTBB          ; SORT THE BAD BLOCKS
004731                                  POP     <R5,R4>         ; RESTORE R5, R4
004731  104265                                                                  MOV (SP)+,R5
004732  104264                                                                  MOV (SP)+,R4
004733  117404                  3$:     DEC     R4              ; DECREMENT COUNT
004734  054715                          BNE     4$              ; IF INCOMPLETE, BRANCH
004735                                  POP     <R5,R4>         ; RESTORE R5, R4
004735  104265                                                                  MOV (SP)+,R5
004736  104264                                                                  MOV (SP)+,R4
004737  114001                          CLR     R1              ; IMMIDATE CALL TO NEXT MODULE
004740  114002                          CLR     R2              ; NO ERRORS
004741  104207  000033                 MOV     #SCHAR0,R0      ; SCHAR0 NEXT MODULE
004743  003231                          BR      JMPRET
                                        .DSABL  LSB
```

```
                                        .SBTTL  SORTBE - SORT THE BEGIN/END SETS IN ASCENDING ORDER
004744                                  SORTBE:
                                        ;
                                        ;       SORT THE BEGIN/END SETS IN ASCENDING ORDER
                                        ;
004744  104672  000016                          MOV     S.BESS+3(R0),R2 ; SEE IF ONLY ONE BEGIN END SET
004746  075003                                  BMI     4$              ; IF SO, EXIT (ALLREADY SORTED)
004747  104202  000013                          MOV     #S.BESS,R2      ; R2 WILL POINT TO START OF BEGIN/END SETS
004751  105072                                  ADD     R0,R2           ; R2 POINTS TO BEGIN/END SETS
004752  104023                          1$:     MOV     R2,R3           ; R3 WILL POINT TO NEXT BEGIN/END SET
004753  105203  000004                  2$:     ADD     #4,R3           ; R3 POINTS TO BEGIN/END SETS
                                        ;
                                        ;       28 BIT COMPARE FOR BEGIN/END SET SORTING
                                        ;
004755  104634  000003                          MOV     3(R3),R4        ; GET WORD THAT R3 POINTS TO
004757  103204  170000                          BIC     #^CHBHINB,R4    ; STRIP OFF UNUSED BITS
004761  104625  000003                          MOV     3(R2),R5        ; GET OTHER WORD TO COMPARE
004763  106045                                  CMP     R4,R5           ; COMPARE
004764  054771                                  BNE     10$             ; IF DIFFERENCE IS FOUND, BRANCH
004765  104625  000002                          MOV     2(R2),R5        ; GET OTHER WORD TO COMPARE
004767  106635  000002                          CMP     2(R3),R5        ; COMPARE
004771  044773                          10$:    BCC     3$              ; IF R2->B/E <= R3->B/E THEN ALLREADY IN ORDER
004772  025004                                  CALL    SWAPBE          ; SWAP TE BEGIN/END SETS
004773  104635  000003                  3$:     MOV     3(R3),R5        ; SEE IF R3->END-OF-LIST
004775  034753                                  BPL     2$              ; IF NOT, BRANCH
004776  105202  000004                          ADD     #4,R2           ; R2->NEXT BEGIN/END SET
005000  104625  000003                          MOV     3(R2),R5        ; SEE IF R2->END-OF-LIST
005002  034752                                  BPL     1$              ; IF NOT, BRANCH
005003  000000                          4$:     RETURN                  ; RETURN TO COPYSU
```

```
                                .SBTTL  SWAPBE - IF BEGIN/END SETS OUT OF ORDER, SWAP
005004                          SWAPBE:
                                ;
                                ;       SWAP THE BEGIN/END SET POINTED TO BY R2 WITH THE BEGIN/END SE
                                ;       POINTED TO BY R3, KEEPING THE END-OF-LIST FLAG INTACT
                                ;
005004                                  PUSH    <R1,R2,R3>      ; SAVE POINTERS
005004  100461                                                                  MOV R1,-(SP)
005005  100462                                                                  MOV R2,-(SP)
005006  100463                                                                  MOV R3,-(SP)
005007  104201  000003                  MOV     #3,R1           ; SET UP LOOP COUNT
005011  104124          1$:             MOV     (R2),R4         ; GET WORD FROM SET
005012  104135                          MOV     (R3),R5         ; GET WORD FROM OTHER SET
005013  100234                          MOV     R4,(R3)+        ; SWAP WORD
005014  100225                          MOV     R5,(R2)+        ; SWAP WORD
005015  117401                          DEC     R1              ; DECREMENT COUNT
005016  055011                          BNE     1$              ; LOOP IF INCOMPLETE
005017  104124                          MOV     (R2),R4         ; GET WORD FROM SET
005020  104135                          MOV     (R3),R5         ; GET WORD FROM OTHER SET
005021  104051                          MOV     R5,R1           ; MOVE TO TEMP STORAGE
005022  103205  177400                  BIC     #HIBYTE,R5      ; STRIP OFF END-OF-LIST FLAG, IF ANY
005024  107051                          SUB     R5,R1           ; R1 CONTAINS END-OF-LIST FLAG, IF ANY
005025  101014                          BIS     R1,R4           ; R4 NOW HAS END-OF-LIST FLAG, IF ANY
005026  100134                          MOV     R4,(R3)         ; SWAP WORD (AND EOL FLAG, IF ANY)
005027  100125                          MOV     R5,(R2)         ; SWAP WORD
005030                                  POP     <R3,R2,R1>      ; RESTORE THE REGISTERS
005030  104263                                                                  MOV (SP)+,R3
005031  104262                                                                  MOV (SP)+,R2
005032  104261                                                                  MOV (SP)+,R1
005033  000000                          RETURN                  ; RETURN TO SORTBE
```

```
                                        .SBTTL  SORTBB - SORT THE BAD BLOCKS IN ASCENDING ORDER
005034                                  SORTBB:
                                        ;
                                        ;       SORT THE BAD BLOCKS IN ASCENDING ORDER
                                        ;
005034  104672  000012                  MOV     S.BADP(R0),R2   ; GET THE BAD BLOCK POINTER
005036  015102                          BEQ     4$              ; IF NO BAD BLOCKS, BRANCH
005037  104625  000001                  MOV     1(R2),R5        ; SEE IF ONLY ONE BAD BLOCK
005041  075102                          BMI     4$              ; IF SO BRANCH (ALLREADY SORTED)
005042  104023                  1$:     MOV     R2,R3           ; R3 WILL POINT AT NEXT BAD BLOCK
005043  105203  000002          2$:     ADD     #2,R3           ; R3 POINTS TO NEXT BAD BLOCK
005045  024632                          CALL    CBB2            ; COMPARE THE TWO BLOCKS
005046  045072                          BCC     3$              ; IF IN ASCENDING ORDER, BRANCH
                                        .SBTTL  SWAPBB - IF BAD BLOCKS OUT OF ORDER, SWAP
                                        ;SWAPBB
                                        ;
                                        ;       SWAP THE BAD BLOCKS, RETAINING THE END-OF-LIST POINTER AT THE END
                                        ;
005047                                  PUSH    R1              ; SAVE R1
005047  100461                                                                  MOV R1,-(SP)
005050  104124                          MOV     (R2),R4         ; GET LO ORDER BAD BLOCK OF 1ST SET
005051  104135                          MOV     (R3),R5         ; GET LO ORDER BAD BLOCK OF 2ND SET
005052  100134                          MOV     R4,(R3)         ; SWAP
005053  100125                          MOV     R5,(R2)         ; SWAP
005054  104624  000001                  MOV     1(R2),R4        ; GET HI ORDER BAD BLOCK OF 1ST SET
005056  104635  000001                  MOV     1(R3),R5        ; GET HI ORDER BAD BLOCK OF 2ND SET
005060  104051                          MOV     R5,R1           ; MOVE TO R1
005061  103205  177400                  BIC     #HIBYTE,R5      ; STRIP OFF END-OF-LIST FLAG, IF ANY
005063  107051                          SUB     R5,R1           ; R1 CONTAINS END-OF-LIST FLAG, IF ANY
005064  101014                          BIS     R1,R4           ; PUT END-OF-LIST FLAG IN R4, IF ANY
005065  100634  000001                  MOV     R4,1(R3)        ; SWAP
005067  100625  000001                  MOV     R5,1(R2)        ; SWAP
005071                                  POP     R1              ; RESTORE R1
005071  104261                                                                  MOV (SP)+,R1
005072  104635  000001          3$:     MOV     1(R3),R5        ; SEE IF END-OF-LIST
005074  035043                          BPL     2$              ; IF NOT, BRANCH
005075  105202  000002                  ADD     #2,R2           ; R2 POINTS TO NEXT BAD BLOCK
005077  104625  000001                  MOV     1(R2),R5        ; SEE IF END-OF-LIST
005101  035042                          BPL     1$              ; IF NOT, BRANCH
005102  000000                  4$:     RETURN                  ; RETURN TO COPYSU
```

```
                                        .SBTTL  SORTTG - SORT THE TRACK/GROUPS IN ASCENDING ORDER
005103                                  SORTTG:
                                        ;
                                        ;       SORTTG WILL SORT THE TRACKS OR GROUPS IN ASCENDING ORDER
                                        ;
005103  104201  000017                  MOV     #S.TGSS,R1      ; R1 WILL POINT TO TRACK/GROUP START
005105  105071                          ADD     R0,R1           ; R1 POINTS TO TRACK/GROUP START
005106  104115                  1$:     MOV     (R1),R5         ; GET START OF LIST
005107  075130                          BMI     4$              ; IF NEGATIVE, WHOLE LIST IS SORTED, EXIT
005110  104012                          MOV     R1,R2           ; R2 WILL POINT TO NEXT LIST MEMBER
005111  115402                  2$:     INC     R2              ; R2 POINTS TO NEXT MEMBER
005112  104125                          MOV     (R2),R5         ; GET NEXT MEMBER
005113  104054                          MOV     R5,R4           ; COPY TO R4
005114  103205  177400                  BIC     #HIBYTE,R5      ; CLEAR END-OF-LIST FLAG (IF ANY)
005116  107054                          SUB     R5,R4           ; SAVE END-OF-LIST FLAG (IF ANY)
005117  106115                          CMP     (R1),R5         ; SEE IF THE MEMBERS ARE IN ORDER
005120  075124                          BMI     3$              ; IF SO, BRANCH
005121  101114                          BIS     (R1),R4         ; COPY START OF LIST TO R4, RETAINING EOL FLAG
005122  100115                          MOV     R5,(R1)         ; SWAP
005123  100124                          MOV     R4,(R2)         ; SWAP
005124  104125                  3$:     MOV     (R2),R5         ; GET MEMBER THAT SORT POINTER POINTS TO
005125  035111                          BPL     2$              ; IF NOT END-OF-LIST, BRANCH
005126  115401                          INC     R1              ; POINT TO NEXT START OF LIST
005127  005106                          BR      1$              ; LOOP
005130  000000                  4$:     RETURN                  ; RETURN TO CALLING PROGRAM
                                        .IF     LE,MAXADR-.
                                MAXADR  =       .+1
                                        .ENDC
```

********** OVERLAY MODULE SCHAR0 - SET UP SUBUNITS, CHECK THAT

```
                                .SBTTL  ********** OVERLAY MODULE SCHAR0 - SET UP SUBUNITS, CHECK THAT ALL PARAMETERS ARE WI
                                        DMOVLY  S0,AREAI
005131  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
000033                          SCHAR0  =       SORT+1
                                ;
                                ;       INSCHR WILL GET THE SUBUNIT CHARACTERISTICS AND INITILIZE THE
                                ;       SUBUNIT PARAMETERS
                                ;
                                        .ENABL  LSB
004710                                  PUSH    R4              ; SAVE R4 (SUBUNIT) POINTER
004710  100464                                                                  MOV R4,-(SP)
004711  104041                          MOV     R4,R1           ; R1 POINTS TO SUBUNIT
004712  104657  000050                  MOV     U.SUBU(R5),R0   ; GET SUBUNIT OFFSET
                                .SBTTL  SMASK  - CALCULATE THE SUBUNIT MASK
                                ;SMASK
                                ;
                                ;       SMASK TAKES THE UNIT OFFSET (0 - 3) IN R0 AND CHANGES IT TO THE
                                ;       SUBUNIT MASK (0001 - 1000)
                                ;
004714                                  PUSH    R1              ; SAVE R1
004714  100461                                                                  MOV R1,-(SP)
004715  104071                          MOV     R0,R1           ; MOVE R0 TO R1
004716  104207  000020                  MOV     #20,R0          ; SUBUNIT 0 MASK
004720  115001                          TST     R1              ; SEE IF SUBUNIT MASK SHIFTED TO CORRECT POSITION
004721  014725                  SMASKL: BEQ     SMASKX          ; IF SO, BRANCH
004722  110207                          ROL     R0              ; SHIFT MASK
004723  117401                          DEC     R1              ; DECREMENT COUNT
004724  004721                          BR      SMASKL          ; BRANCH
004725                          SMASKX: POP     R1              ; RESTORE R1
004725  104261                                                                  MOV (SP)+,R1
004726  104070  004504                  MOV     R0,SUBUNT       ; MOVE TO SDI BUFFER
004730  104203  004467                  MOV     #CR.SCR,R3      ; POINT TO COMMAND
004732                                  PUSH    R1              ; SAVE R1
004732  100461                                                                  MOV R1,-(SP)
004733  021173                          CALL    TALK            ; SDI EXCHANGE
004734                                  POP     R1              ; RESTORE R1
004734  104261                                                                  MOV (SP)+,R1
004735  115002                          TST     R2              ; SEE IF ERROR OCCURRED
004736  055144                          BNE     13$             ; IF SO, BRANCH
004737  100612  000010                  MOV     R2,S.MEGR(R1)   ; ZERO MEGABIT COUNT
004741  100612  000011                  MOV     R2,S.MEGW(R1)   ; ZERO MEGABIT COUNT
004743  104203  001702                  MOV     #ST,R3          ; R3 POINTS TO SUBUNIT CHARACTERISTCS
004745  106205  007702                  CMP     #FIRSTU,R5      ; IS R5 -> FIRST UNIT?
004747  014763                          BEQ     1$              ; IF SO, BRANCH
004750  104202  007702                  MOV     #FIRSTU,R2      ; R2 -> FIRST UNIT
004752                                  PUSH    <R1,R3,R4>      ; SAVE REGS
004752  100461                                                                  MOV R1,-(SP)
004753  100463                                                                  MOV R3,-(SP)
004754  100464                                                                  MOV R4,-(SP)
004755  025165                          CALL    TRAV            ; TRAVERSE THE UNITS TO FIND IF SUBUNIT CHAR ARE EQUAL
```

```
     49 004756                                      POP     <R4,R3,R1>      ; RESTORE REGS
        004756  104264                                                              MOV (SP)+,R4
        004757  104203                                                              MOV (SP)+,R3
        004760  104261                                                              MOV (SP)+,R1
     50 004761  115007                              TST     R0              ; WERE ANY CHARS EQUAL?
     51 004762  054776                              BNE     3$              ; IF SO, BRANCH (R0 WILL be NONE ZERO IF MATCH)
     52 004763  104207  000023           1$:        MOV     #19.,R0         ; MOVE WORD COUNT TO R0
     53 004765                                      PUSH    R0              ; SAVE COUNT
        004765  100467                                                              MOV R0,-(SP)
     54 004766  024625                              CALL    GETMEM          ; GET SOME MEMORY
     55 004767  100617  000007                      MOV     R0,S.SCHR(R1)   ; STORE POINTER TO IN SUBUNIT PARAMETERS
     56 004771                                      POP     R2              ; RESTORE COUNT IN R2
        004771  104262                                                              MOV (SP)+,R2
     57 004772  104234                   2$:        MOV     (R3)+,R4        ; MOVE ONE WORD OF S CHAR TO R0
     58 004773  100274                              MOV     R4,(R0)+        ; MOVE TO SUBUNIT CHAR AREA
     59 004774  117402                              DEC     R2              ; DECREMENT WORD COUNT
     60 004775  054772                              BNE     2$              ; IF COUNT UNEXPIRED, BRANCH
     61 004776  104117                   3$:        MOV     (R1),R0         ; GET SUBUNIT PARAMETERS
     62 004777                                      ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
     63 004777  102207  020000                      BIT     #DCYLS,R0       ; SEE IF DIAGNOSTIC CYLINDERS ARE USED
     64 005001  015011                              BEQ     4$              ; IF NOT, BRANCH
     65 005002  104202  005524                      MOV     #D$,R2          ; MOVE THE CHARACTER 'D' (TO MAKE DBN) TO R3
     66 005004  104303  001706                      MOV     ST+RBNTRK,R3    ; GET RBN'S PER TRACK
     67 005006  103203  177600                      BIC     #HIBYTE!200,R3  ; CLEAR UNUSED BITS
     68 005010  005014                              BR      5$              ; BRANCH
     69 005011  104202  005522           4$:        MOV     #L$,R2          ; MOVE THE CHARACTER 'L' (TO MAKE LBN) TO R3
     70 005013  114003                              CLR     R3              ; FOR LBN'S, NO RBN'S PER TRACK ADDED
     71 005014  100612  000005           5$:        MOV     R2,S.LETR(R1)   ; SAVE
     72 005016  105303  001713                      ADD     ST+LBNTRK,R3    ; ADD LBNS PER TRACK (TO ZERO IF LBN AREA)
     73 005020  103203  177400                      BIC     #HIBYTE,R3      ; CLEAR UNUSED BITS
     74 005022  104030  004513                      MOV     R3,SECTRK       ; SAVE IN SECTORS PER TRACK
     75 005024  100613  000006                      MOV     R3,S.TRKL(R1)   ; SAVE IN TRACK LENGTH
     76 005026  025717                              CALL    COMPSC          ; COMPUTE SECTORS/GROUP AND SECTORS/CYL
     77 005027  025746                              CALL    CLCMAX          ; CALCULATE MAXIMUM DBN NUMBER (IN CASE NEEDED)
     78 005030  104117                              MOV     (R1),R0         ; GET SUBUNIT PARAMETERS
     79 005031                                      ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
     80 005031  102207  000040                      BIT     #BEUSED,R0      ; SEE IF BEGIN/END SETS ARE USED
     81 005033  015037                              BEQ     6$              ; IF NOT, BRANCH
     82 005034  102207  000200                      BIT     #ONLYCL,R0      ; SEE IF WE ALLREADY HAVE BEGIN/END SETS
     83 005036  015042                              BEQ     7$              ; IF SO, BRANCH
     84 005037  025243                   6$:        CALL    CHKCYL          ; CONVERT THE CYLS TO BN'S
     85 005040  115002                              TST     R2              ; SEE IF AN ERROR OCCURRED
     86 005041  055065                              BNE     9$              ; IF SO, BRANCH
     87 005042  025420                   7$:        CALL    CHKBES          ; CHECK THE BEGIN/END SETS FOR ERRORS
     88 005043  115002                              TST     R2              ; SEE IF AN ERROR OCCURRED
     89 005044  055065                              BNE     9$              ; IF SO, BRANCH
     90 005045  104302  001702                      MOV     ST+LBNCYL,R2    ; R2 CONTAINS LO LBN CYLS
     91 005047  105302  001723                      ADD     ST+XBNCYL,R2    ; ADD LO XBN CYLS TO R2
     92 005051  100612  000002                      MOV     R2,S.SDCL(R1)   ; MOVE TO LO STARTING DIAG CYL
     93 005053  104302  001703                      MOV     ST+LBNCYL+1,R2  ; GET HI ORDER LBN CYLS
     94 005055  045057                              BCC     8$              ; IF NO CARRY, BRANCH
     95 005056  115402                              INC     R2              ; PROPOGATE CARRY
     96 005057  100612  000003           8$:        MOV     R2,S.SDCL+1(R1) ; STORE HI STARTING DIAG CYL
     97 005061  114002                              CLR     R2              ; NO ERRORS
     98 005062  104207  000034                      MOV     #SCHAR1,R0      ; SCHAR1 IS NEXT MODULE
     99 005064  005162                              BR      15$             ; EXIT
    100 005065  104650  000063  001106   9$:        MOV     U.UNUM(R5),HRQ.03       ; GET STARTING UNIT NUMBER
```

```
                                                ADD     U.SUBU(R5),HRQ.03       ; ADD OFFSET
                                                MOV     HRQ.RQ,R0               ; SET UP TO REPORT ERROR
                                                CALL    HOSTRQ                  ; REPORT
                                                BIS     #DIE,M.PARM             ; FLAG INITIALIZATION ERROR
                                                MOV     #U.SUBP,R1              ; R1 WILL POINT AT SUBUNIT POINTERS
                                                ADD     R5,R1                   ; R1 POINTS AT SUBUNIT POINTERS
                                                ADD     U.SUBU(R5),R1           ; R1 POINTS AT POINTER TO SUBUNIT BEING HANDLED
                                                MOV     (R1),R0                 ; R0 POINTS TO SUBUNIT BEGING HANDLED
                                                MOV     #DROP,R3                ; MOVE DROP FLAG TO R3
                                                MOV     R3,(R0)                 ; DROP THIS SUBUNIT
                                                ASSUME  S.PARM,0
                                                CLR     R2                      ; NO ERRORS
                                        10$:    MOV     U.SUBU(R5),R3           ; GET SUBUNIT OFFSET
                                                INC     R3                      ; NEXT SUBUNIT
                                                MOV     R3,U.SUBU(R5)           ; SAVE
                                                CMP     #3,R3                   ; SEE IF VALID OFFSET
                                                BCS     11$                     ; IF NOT, BRANCH
                                                                                                BCC     .+2
                                                                                                BR      11$
                                                ADD     #U.SUBP,R3              ; WILL POINT TO NEXT SUBUNIT POINTER
                                                ADD     R5,R3                   ; ADD POINTER TO UNIT DATABASE
                                                MOV     (R3),R3                 ; GET POINTER
                                                BMI     10$                     ; IF NO SUBUNIT, BRANCH
                                                MOV     (R3),R1                 ; GET SUBUNIT PARAMETERS
                                                ASSUME  S.PARM,0
                                                BMI     10$                     ; IF DROPPED, BRANCH
                                                ASSUME  DROP,100000
                                                BR      12$                     ; EXIT
                                        11$:    CLR     R3                      ; NO SUBUNITS
                                                MOV     R3,U.SUBU(R5)           ; SAVE
                                        12$:    TST     R3                      ; ANY?
                                                BNE     14$                     ; IF SO, BRANCH
                                                MOV     #GETSER,R0              ; GETSER NEXT MODULE
                                                BR      15$                     ; EXIT
                                        13$:    CERROR  5,#SER5                 ; REPORT SECONDARY ERROR
                                                                                        MOV     #SER5,HRQ.05
                                                BIC     #C2DFTL,HRQ.02          ; CHANGE ERROR TO HARD ERROR
                                                MOV     #ERRMC,HRQ.RQ           ; COUNT ERROR
                                                BIS     #DIE,M.PARM             ; FLAG INITIALIZATION ERROR
                                        14$:    MOV     #SCHAR0,R0              ; THIS MODULE IS NEXT
                                        15$:    CLR     R1                      ; IMMIDATE CALL TO NEXT MODULE
                                                POP     R4                      ; RESTORE R4
                                                                                                MOV (SP)+,R4
                                                BR      JMPRET                  ; RETURN TO             141
```

```
                                        .SBTTL  TRAV - TRAVERSE THOUGH THE UNITS TO FIND SUBUNIT CHARS THAT MATCH
005165                          TRAV:   PUSH    R2                      ; SAVE R2 -> UNIT
005165  100462                                                                          MOV R2,-(SP)
005166  025176                          CALL    SUBTRV                  ; GO TRAVERSE SUBUNITS
005167                                  POP     R2                      ; RESTORE R2
005167  104262                                                                          MOV (SP)+,R2
005170  115007                          TST     R0                      ; FOUND A MATCH
005171  055175                          BNE     1$                      ; IF SO, BRANCH
005172  104122                          MOV     (R2),R2                 ; ELSE, CHECK NEXT UNIT
005173  106052                          CMP     R5,R2                   ; ARE WE POINTING TO SAME UNIT?
005174  055165                          BNE     TRAV                    ; IF NOT, CONTINUE
005175  000000                  1$:     RETURN                          ; ELSE, EXIT

                                        .SBTTL  SUBTRV - TRAVERSE THROUGH SUBUNITS TO FIND SUBUNIT CHARS THAT MATCH
005176  104204  000004            SUBTRV: MOV     #4,R4                   ; R1 IS COUNTER TO END
005200  115402                          INC     R2                      ; R2 -> SUBUNIT PARAMETER POINTER
005201                                  ASSUME  U.SUBP,1
005201  104223                  1$:     MOV     (R2)+,R3                ; R3 HAS ADDRESS OF SUBUNIT PARAMETER
005202  075214                          BMI     2$                      ; IF MINUS, NOT A GOOD SUBUNIT
005203  104637  000007                  MOV     S.SCHR(R3),R0           ; R0 -> SUBUNIT CHAR
005205                                  PUSH    <R4,R2>
005205  100464                                                                          MOV R4,-(SP)
005206  100462                                                                          MOV R2,-(SP)
005207  025217                          CALL    SCHRCP                  ; CHECK SUBUNIT CHARACTERISTICS
005210                                  POP     <R2,R4>
005210  104262                                                                          MOV (SP)+,R2
005211  104264                                                                          MOV (SP)+,R4
005212  115007                          TST     R0                      ; FIND A MATCH?
005213  055216                          BNE     3$                      ; IF SO, EXIT
005214  117404                  2$:     DEC     R4                      ; ELSE, ARE ALL SUBUNITS DONE FOR THIS UNIT?
005215  055201                          BNE     1$                      ; IF NOT, BRANCH
005216  000000                  3$:     RETURN

                                        .SBTTL  SCHRCP - SUBUNIT CHARACTERISTICS COMPARE
                                        ; *** R0 = 0 IF NO MATCH OR R0 -> SUBUNIT CHAR IF THERE IS A MATCH
005217                          SCHRCP: PUSH    <R0,R1>
005217  100467                                                                          MOV R0,-(SP)
005220  100461                                                                          MOV R1,-(SP)
005221  104204  001702                  MOV     #ST,R4                  ; R4 -> CHARACTERISTICS
005223  104201  000023                  MOV     #19.,R1                 ; R1 = # OF WORDS TO COMPARE
005225  104242                  1$:     MOV     (R4)+,R2                ; COMPARE CHARACTERISTICS
005226  106272                          CMP     (R0)+,R2                ; EQUAL?
005227  055237                          BNE     2$                      ; IF NOT, EXIT
005230  117401                          DEC     R1                      ; ELSE, CHECK NEXT WORD
005231  055225                          BNE     1$                      ; IF NOT DONE WITH COMPARISON, BRANCH
005232                                  POP     <R1,R0>                 ; RESTORE REGS
005232  104261                                                                          MOV (SP)+,R1
005233  104267                                                                          MOV (SP)+,R0
005234  100617  000007                  MOV     R0,S.SCHR(R1)           ; AND STORE POINT TO CHAR INTO SUBUNIT PARAM
005236  005242                          BR      3$                      ; AND EXIT WITH MATCH
005237                          2$:     POP     <R1,R0>
005237  104261                                                                          MOV (SP)+,R1
005240  104267                                                                          MOV (SP)+,R0
005241  114007                          CLR     R0                      ; R0 = 0 (NO MATCH)
005242  000000                  3$:     RETURN
```

```
                                .SBTTL  CHKCYL - CHECK VALIDITY OF STARTING AND ENDING CYLS, CONVERT TO BEGIN/END SET
005243                          CHKCYL:
                                ;
                                ;       CALCULATE THE STARTING AND ENDING BN'S FROM THE GIVEN STARTING
                                ;       AND ENDING CYLS
                                ;
005243                          PUSH    R5              ; SAVE R4, R5
005243  100465                                                          MOV R5,-(SP)
005244  104614  000015          MOV     S.BESS+2(R1),R4 ; R4 HAS LO STARTING CYL
005246  104615  000016          MOV     S.BESS+3(R1),R5 ; R5 HAS HI STARTING CYL
005250  103205  100000          BIC     #100000,R5      ; CLEAR EOL FLAG (IF ANY)
005252  025346                  CALL    CYLBN           ; CALCULATE STARTING BN ON THAT CYL
005253  115002                  TST     R2              ; SEE IF ANY ERROR OCCURRED
005254  055344                  BNE     10$             ; IF SO, BRANCH
005255  100617  000015          MOV     R0,S.BESS+2(R1) ; SAVE LO ORDER STARTING BN
005257  101203  100000          BIS     #100000,R3      ; SET END-OF-LIST FLAG
005261  100613  000016          MOV     R3,S.BESS+3(R1) ; SAVE HI ORDER STARTING BN
005263  104615  000014          MOV     S.BESS+1(R1),R5 ; GET HI ORDER ENDING CYL
005265  106205  177777          CMP     #-1,R5          ; SEE IF SHOULD DEFAULT TO HIGHEST LBN
005267  015312                  BEQ     4$              ; IF SO, BRANCH
005270  104614  000013          MOV     S.BESS(R1),R4   ; GET LO ORDER ENDING CYL
005272  105204  000001          ADD     #1,R4           ; FIND STARTING BN OF NEXT CYL
005274  045276                  BCC     2$              ; IF NO CARRY, BRANCH
005275  115405                  INC     R5              ; PROPOGATE CARRY
005276  025346          2$:     CALL    CYLBN           ; CACULATE STARTING BN OF NEXT CYL
005277  115002                  TST     R2              ; SEE IF ANY ERRORS
005300  055344                  BNE     10$             ; IF SO, BRANCH
005301  107207  000001          SUB     #1,R0           ; GET LAST BN OF PREVIOUS CYL
005303  045305                  BCC     3$              ; IF NO CARRY, BRANCH
005304  117403                  DEC     R3              ; PROPOGATE CARRY
005305  100617  000013          3$:     MOV     R0,S.BESS(R1)   ; SAVE LO ORDER ENDING BN
005307  100613  000014          MOV     R3,S.BESS+1(R1) ; SAVE HI ORDER ENDING BN
005311  005343                  BR      9$              ; EXIT
005312  104117          4$:     MOV     (R1),R0         ; GET SUBUNIT PARAMETERS
005313                          ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
005313  102207  020000          BIT     #DCYLS,R0       ; SEE IF DIAGNOSTIC CYLINDERS USED
005315  055333                  BNE     6$              ; IF SO, BRANCH
005316  104307  001714          MOV     ST+LBNHST,R0    ; GET LO ORDER ENDING LBN
005320  107207  000001          SUB     #1,R0           ; LAST LBN IN LBN AREA
005322  100617  000013          MOV     R0,S.BESS(R1)   ; SAVE LO ORDER
005324  104307  001715          MOV     ST+LBNHST+1,R0  ; GET HI ORDER
005326  045330                  BCC     5$              ; IF NO CARRY, BRANCH
005327  117407                  DEC     R0              ; PROPOGATE CARRY
005330  100617  000014          5$:     MOV     R0,S.BESS+1(R1) ; SAVE HI ORDER
005332  005343                  BR      9$              ; EXIT
005333  104307  004511          6$:     MOV     MAXDBN,R0       ; GET LO ORDER MAX DBN
005335  100617  000013          MOV     R0,S.BESS(R1)   ; SAVE
005337  104307  004512          MOV     MAXDBN+1,R0     ; GET HI ORDER
005341  100617  000014          MOV     R0,S.BESS+1(R1) ; SAVE
005343  114002          9$:     CLR     R2              ; NO ERRORS
005344                  10$:    POP     R5              ; RESTORE
005344  104265                                                          MOV (SP)+,R5
005345  000000                  RETURN
```

```
                                        .SBTTL  CYLBN  - GIVEN A CYLINDER, CALCULATE STARTING BN ON THAT CYLINDER
005346                                  CYLBN:
                                        ;
                                        ;       TAKING THF CYL GIVEN IN R4,R5 CALCULATE THE FIRST BN ON THAT
                                        ;       CYL AND RETURN IN R2,R3.  IF BN EXCEEDS 28 BITS, REPORT
                                        ;       ERROR
                                        ;
005346  114007                          CLR     R0              ; CLEAR THE PRODUCT AREA
005347  114003                          CLR     R3              ; CLEAR THE PRODUCT AREA
005350  115004                          TST     R4              ; SEE IF LO ORDER IS ZERO
005351  055354                          BNE     1$              ; IF NOT, BRANCH
005352  115005                          TST     R5              ; SEE IF HI ORDER IS ZERO
005353  015416                          BEQ     5$              ; IF SO, BRANCH AND EXIT
005354  107204  000001          1$:     SUB     #1,R4           ; ADJUST COUNT
005356  045360                          BCC     2$              ; IF NO BORROW, BRANCH
005357  117405                          DEC     R5              ; PROPOGATE BORROW
005360  105307  004515          2$:     ADD     SECCYL,R0       ; ADD LO SECTORS/CYL TO LO ORDER PROD
005362  045364                          BCC     3$              ; IF NO CARRY, BRANCH
005363  115403                          INC     R3              ; PROPOGATE CARRY
005364  106203  007777          3$:     CMP     #HBHINB,R3      ; SEE IF HI ORDER TOO BIG
005366  045410                          BCC     4$              ; IF NOT, BRANCH
005367                                  DEVFTL  57,#SER20       ; REPORT LBN OVERFLOW
005367  104200  003647  001107                                                  MOV     #ER57,HRQ.04
005372  104200  004052  001110                                                  MOV     #SER20,HRQ.05
005375  104202  047731                                                          MOV     #57!FTLDEV+4000.,R2
005377  104020  001105                                                          MOV     R2,HRQ.02
005401  104200  005401  001104                                                  MOV     #.,HRQ.01
005404  104200  060014  001103                                                  MOV     #ERRMC,HRQ.RQ
005407  005417                          BR      6$              ; EXIT
005410  107204  000001          4$:     SUB     #1,R4           ; DECREMENT LO ORDER COUNT
005412  045360                          BCC     2$              ; IF INCOMPLETE, BRANCH
005413  107205  000001                  SUB     #1,R5           ; DECREMENT HI ORDER COUNT
005415  045360                          BCC     2$              ; IF INCOMPLETE, BRANCH
005416  114002                  5$:     CLR     R2              ; NO ERRORS
005417  000000                  6$:     RETURN                  ; RETURN TO CALLING PROGRAM
```

```
      1                                 .SBTTL  CHKBES - CHECK THE BEGIN/END SETS FOR ERRORS
      2 005420                          CHKBES:
      3                                 ;
      4                                 ;       CHECK THAT THE BEGIN BLOCK <= END BLOCK IN EACH BEGIN/END SET,
      5                                 ;       THAT THE END BLOCK < THE BEGIN BLOCK OF THE NEXT BEGIN/END SET,
      6                                 ;       AND THAT THE LAST END BLOCK IS A VALID LBN OR DBN FOR THAT DRIVE.
      7                                 ;
      8 005420                                  PUSH    R5
        005420  100465                                                                          MOV R5,-(SP)
      9 005421  104203  000013                  MOV     #S.BESS,R3      ; R3 WILL POINT TO ENDING BLOCK NUMBER
     10 005423  105013                          ADD     R1,R3           ; R3 POINTS TO ENDING BN
     11 005424  104202  000015                  MOV     #S.BESS+2,R2    ; R2 WILL POINT TO STARTING BN
     12 005426  105012                          ADD     R1,R2           ; R2 POINTS TO STARTING BN
     13 005427  024632                  1$:     CALL    CBB2            ; COMPARE
     14 005430  045502                          BCC     2$              ; IF NO ERROR, BRANCH
     15 005431  104112                          MOV     (R1),R2         ; GET SUBUNIT PARAMETERS
     16 005432                                  ASSUME  S.PARM,0
     17 005432  102202  000040                  BIT     #BEUSED,R2      ; SEE IF BEGIN/END SETS USED
     18 005434  015440                          BEQ     20$             ; IF NOT, BRANCH
     19 005435  102202  000200                  BIT     #ONLYCL,R2      ; SEE IF ONLY THE CYLINDER IS SPECIFIED
     20 005437  015461                          BEQ     21$             ; IF NOT, BRANCH
     21 005440                          20$:    DEVFTL  55,#SER20       ; REPORT BEGIN CYL > ENDING CYL
        005440  104200  003555  001107                                          MOV     #ER55,HRQ.04
        005443  104200  004052  001110                                          MOV     #SER20,HRQ.05
        005446  104202  047727                                                  MOV     #55!FTLDEV+4000.,R2
        005450  104020  001105                                                  MOV     R2,HRQ.02
        005452  104200  005452  001104                                          MOV     #.,HRQ.01
        005455  104200  060014  001103                                          MOV     #ERRMC,HRQ.RQ
     22 005460  005715                          BR      7$              ; BRANCH TO ERROR EXIT
     23 005461                          21$:    DEVFTL  50,#SER20       ; REPORT BEGIN > ENDING BN
        005461  104200  003320  001107                                          MOV     #ER50,HRQ.04
        005464  104200  004052  001110                                          MOV     #SER20,HRQ.05
        005467  104202  047722                                                  MOV     #50!FTLDEV+4000.,R2
        005471  104020  001105                                                  MOV     R2,HRQ.02
        005473  104200  005473  001104                                          MOV     #.,HRQ.01
        005476  104200  060014  001103                                          MOV     #ERRMC,HRQ.RQ
     24 005501  005715                          BR      7$              ; BRANCH TO ERROR EXIT
     25 005502  104625  000001          2$:     MOV     1(R2),R5        ; SEE IF THE END-OF-LIST HAS BEEN FOUND
     26 005504  075536                          BMI     4$              ; IF SO, BRANCH
     27 005505  105202  000004                  ADD     #4,R2           ; POINT TO NEXT BEGIN SET
     28 005507  024632                          CALL    CBB2            ; COMPARE
     29 005510                                  BCS     3$              ; BRANCH
        005510  045512                                                                          BCC     .+2
        005511  005533                                                                          BR      3$
     30 005512                                  DEVFTL  51,#SER20       ; STARTING BN <= PREVIOUS ENDING BN
        005512  104200  003365  001107                                          MOV     #ER51,HRQ.04
        005515  104200  004052  001110                                          MOV     #SER20,HRQ.05
        005520  104202  047723                                                  MOV     #51!FTLDEV+4000.,R2
        005522  104020  001105                                                  MOV     R2,HRQ.02
        005524  104200  005524  001104                                          MOV     #.,HRQ.01
        005527  104200  060014  001103                                          MOV     #ERRMC,HRQ.RQ
     31 005532  005715                          BR      7$              ; ERROR EXIT
     32 005533  105203  000004          3$:     ADD     #4,R3           ; R3 NOW POINTS TO NEXT END SET
     33 005535  005427                          BR      1$              ; LOOP
     34 005536  104112                  4$:     MOV     (R1),R2         ; GET SUBUNIT CHARACTERISTICS
     35 005537                                  ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
     36 005537  102202  020000                  BIT     #DCYLS,R2       ; SEE IF DIAGNOSTIC CYLINDERS ARE IN USE
```

```
     005541  055622                          BNE     5$                  ; IF SO, BRANCH
     005542  104202  001714                  MOV     #ST+LBNHST,R2       ; R2 POINTS TO LBNS IN HOST AREA
     005544  024632                          CALL    CBB2                ; COMPARE
     005545                                  BCS     6$                  ; EXIT
     005545  045547                                                                      BCC     .+2
     005546  005714                                                                      BR      6$
     005547  104112                          MOV     (R1),R2             ; GET SUBUNIT CHARACTERISTICS
     005550                                  ASSUME  S.PARM,0            ; ASSUME THAT S.PARM IS ZERO
     005550  102202  000040                  BIT     #BEUSED,R2          ; SEE IF TRACKS/GROUPS IN USE
     005552  015556                          BEQ     10$                 ; IF SO, BRANCH
     005553  102202  000200                  BIT     #ONLYCL,R2          ; SEE IF B/E SET COMPUTED BY CYLINDERS
     005555  015565                          BEQ     11$                 ; IF NOT, BRANCH
     005556                          10$:    CERROR  6,#L$               ; LBN ERROR
     005556  104200  005522  001111                                                      MOV     #L$,HRQ.06
     005561                                  CERROR  7,#L$               ; LBN ERROR
     005561  104200  005522  001112                                                      MOV     #L$,HRQ.07
     005564  005644                          BR      14$                 ; EXIT
     005565  107200  000001  001714  11$:    SUB     #1,ST+LBNHST        ; DECREMENT LO LBN COUNT BY 1
     005570  045573                          BCC     9$                  ; IF NO BORROW, BRANCH
     005571  117400  001715                  DEC     ST+LBNHST+1         ; PROPOGATE BORROW
     005573                          9$:     DEVFTL  52,<#SER20,ST+LBNHST,ST+LBNHST+1> ; LAST ENDING BN > MAX
     005573  104200  003410  001107                                                      MOV     #ER52,HRQ.04
     005576  104200  004052  001110                                                      MOV     #SER20,HRQ.05
     005601  104300  001714  001111                                                      MOV     ST+LBNHST,HRQ.06
     005604  104300  001715  001112                                                      MOV     ST+LBNHST+1,HRQ.07
     005607  104202  047724                                                              MOV     #52!FTLDEV+4000.,R2
     005611  104020  001105                                                              MOV     R2,HRQ.02
     005613  104200  005613  001104                                                      MOV     #.,HRQ.01
     005616  104200  060014  001103                                                      MOV     #ERRMC,HRQ.RQ
     005621  005715                          BR      7$                  ; ERROR EXIT
     005622  104032                  5$:     MOV     R3,R2               ; R2 POINTS TO LAST END SET
     005623  104203  004511                  MOV     #MAXDBN,R3          ; R3 POINTS TO MAXIMUM DBN
     005625  024632                          CALL    CBB2                ; COMPARE
     005626  045714                          BCC     6$                  ; EXIT
     005627  104112                          MOV     (R1),R2             ; GET SUBUNIT CHARACTERISTICS
     005630                                  ASSUME  S.PARM,0            ; ASSUME THAT S.PARM IS ZERO
     005630  102202  000040                  BIT     #BEUSED,R2          ; SEE IF TRACKS/GROUPS IN USE
     005632  015636                          BEQ     12$                 ; IF SO, BRANCH
     005633  102202  000200                  BIT     #ONLYCL,R2          ; SEE IF B/E SET COMPUTED BY CYLINDERS
     005635  015665                          BEQ     13$                 ; IF NOT, BRANCH
     005636                          12$:    CERROR  6,#D$               ; DBN ERROR
     005636  104200  005524  001111                                                      MOV     #D$,HRQ.06
     005641                                  CERROR  7,#D$               ; DBN ERROR
     005641  104200  005524  001112                                                      MOV     #D$,HRQ.07
     005644                          14$:    DEVFTL  62,#SER20           ; REPORT ERROR
     005644  104200  003770  001107                                                      MOV     #ER62,HRQ.04
     005647  104200  004052  001110                                                      MOV     #SER20,HRQ.05
     005652  104202  047736                                                              MOV     #62!FTLDEV+4000.,R2
     005654  104020  001105                                                              MOV     R2,HRQ.02
     005656  104200  005656  001104                                                      MOV     #.,HRQ.01
     005661  104200  060014  001103                                                      MOV     #ERRMC,HRQ.RQ
     005664  005715                          BR      7$                  ; EXIT
     005665                          13$:    DEVFTL  52,<#SER20,MAXDBN,MAXDBN+1> ; LAST ENDING BN > MAX
     005665  104200  003410  001107                                                      MOV     #ER52,HRQ.04
     005670  104200  004052  001110                                                      MOV     #SER20,HRQ.05
     005673  104300  004511  001111                                                      MOV     MAXDBN,HRQ.06
     005676  104300  004512  001112                                                      MOV     MAXDBN+1,HRQ.07
```

```
        005701  104202  047724                                          MOV     #52!FTLDEV+4000.,R2
        005703  104020  001105                                          MOV     R2,HRQ.02
        005705  104200  005705  001104                                  MOV     #.,HRQ.01
        005710  104200  060014  001103                                  MOV     #ERRMC,HRQ.RQ
    70  005713  005715                          BR      7$              ; ERROR EXIT
    71  005714  114002                  6$:     CLR     R2              ; FLAG AS NO ERROR
    72  005715                          7$:     POP     R5              ; RESTORE REGISTER
        005715  104265                                                          MOV (SP)+,R5
    73  005716  000000                          RETURN                  ; RETURN TO INSCHR
```

```
      1                                     .SBTTL  COMPSC - COMPUTE SECTORS/GROUP AND SECTORS/CYLINDER
      2 005717                              COMPSC:
      3                                     ;
      4                                     ;       COMPUTE HOW MANY SECTORS/GROUP AND SECTORS/CYLINDER
      5                                     ;       GIVEN HOW MANY SECTORS/TRACK IN SECTRK
      6                                     ;
      7 005717  114007                              CLR     R0              ; CLEAR RUNNING TOTAL
      8 005720  104302  001705                     MOV     ST+TRKGRP,R2    ; GET NUMBER OF TRACKS PER GROUP
      9 005722  103202  177400                     BIC     #HIBYTE,R2      ; CLEAR UNUSED BITS
     10 005724  105307  004513              1$:    ADD     SECTRK,R0       ; ADD NUMBER OF SECTORS/TRACK TO RUNNING TOTAL
     11 005726  117402                              DEC     R2              ; DECREMENT COUNT
     12 005727  055724                              BNE     1$              ; IF COUNT INCOMPLETE, BRANCH
     13 005730  104070  004514                     MOV     R0,SECGRP       ; SAVE SECTORS/GROUP
     14 005732  114007                              CLR     R0              ; CLEAR RUNNING TOTAL
     15 005733  104302  001704                     MOV     ST+GRPCYL,R2    ; GET GROUPS/CYLINDER
     16 005735  103202  177400                     BIC     #HIBYTE,R2      ; CLEAR UNUSED BITS
     17 005737  105307  004514              2$:    ADD     SECGRP,R0       ; ADD SECTORS/GROUP TO RUNNING TOTAL
     18 005741  117402                              DEC     R2              ; DECREMENT COUNT
     19 005742  055737                              BNE     2$              ; IF COUNT INCOMPLETE, BRANCH
     20 005743  104070  004515                     MOV     R0,SECCYL       ; STORE SECTORS/CYLINDER
     21 005745  000000                              RETURN                  ; RETURN TO CALLING PROGRAM
```

```
                                        .SBTTL  CLCMAX - CALCULATE THE MAXIMUM WRITEABLE DBN
005746                                  CLCMAX:
                                        ;
                                        ;       CALCULATE THE MAXIMUM DBN NUMBER ALLOWED, AND PLACE IT IN MAXDBN
                                        ;       FOR BEGIN/END SET GENERATION AND TESTING (BAD BLOCKS TOO)
                                        ;
005746                                          PUSH    R1              ; SAVE REGISTER
005746  100461                                                                  MOV R1,-(SP)
005747  114007                                  CLR     R0              ; CLEAR PRODUCT AREA
005750  114001                                  CLR     R1              ; CLEAR PRODUCT AREA
005751  104302  001724                          MOV     ST+DBNCYL,R2    ; GET DIAG CYL COUNT
005753  110702                                  SWAB    R2              ; MOVE TO LO ORDER BYTE
005754  103202  177400                          BIC     #HIBYTE,R2      ; CLEAR UNUSED BITS
005756  105307  004515                  1$:     ADD     SECCYL,R0       ; ADD SECTORS/CYL TO TOTAL
005760  045762                                  BCC     2$              ; IF NO CARRY, BRANCH
005761  115401                                  INC     R1              ; INCREMENT HI WORD
005762  117402                          2$:     DEC     R2              ; DECREMENT COUNT
005763  055756                                  BNE     1$              ; IF NO CARRY, BRANCH
005764  104302  001724                          MOV     ST+DBNCYL,R2    ; GET NUMBER OF READ ONLY GROUPS
005766  103202  177400                          BIC     #HIBYTE,R2      ; CLEAR UNUSED BITS
005770  107307  004514                  3$:     SUB     SECGRP,R0       ; SUBTRACT NUMBER OF SECTORS/GROUP
005772  045774                                  BCC     4$              ; IF NO BORROW, BRANCH
005773  117401                                  DEC     R1              ; PROPOGATE BORROW
005774  117402                          4$:     DEC     R2              ; DECREMENT COUNT
005775  055770                                  BNE     3$              ; IF INCOMPLETE, BRANCH
005776  107207  000001                          SUB     #1,R0           ; ADJUST FOR DBNS STARTING AT ZERO
006000  046002                                  BCC     5$              ; IF NO BORROW, BRANCH
006001  117401                                  DEC     R1              ; PROPOGATE BORROW
006002  104070  004511                  5$:     MOV     R0,MAXDBN       ; SAVE LO ORDER WORD
006004  104010  004512                          MOV     R1,MAXDBN+1     ; SAVE HI ORDER WORD
006006                                          POP     R1              ; RESTORE REGISTERS
006006  104261                                                                  MOV (SP)+,R1
006007  000000                                  RETURN                  ; RETURN TO INSCHR
                                                .IF     LE,MAXADR-.
                                        MAXADR  =       .+1
                                                .ENDC
```

```
                                .SBTTL  ********** OVERLAY MODULE SCHAR1 - SET UP SUBUNITS, CHECK TRACK/GROUP PARAMETERS
006010                                  DMOVLY  S1,AREAI
006010  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;*******************************************************************************
                                ;
                                ;
000034                          SCHAR1  =       SCHAR0+1
                                ;
                                ;       SCHAR1 WILL INITILIZE THE SUBUNIT PARAMETERS (TRACK/GROUP ONLY)
                                ;
                                ;
                                        .ENABL  LSB
004710                                  PUSH    R4              ; SAVE SUBUNIT POINTER
004710  100464                                                                  MOV R4,-(SP)
004711  104141                          MOV     (R4),R1         ; GET SUBUNIT PARAMETERS
004712                                  ASSUME  S.PARM,0
004712  102201  004000                  BIT     #RONLY,R1       ; SEE IF READ ONLY
004714  014762                          BEQ     5$              ; IF NOT, BRANCH
004715  102201  002000                  BIT     #WONLY,R1       ; SEE IF WRITE ONLY IS SET TOC
004717  014736                          BEQ     6$              ; IF NOT, BRANCH
004720                                  DEVFTL  28              ; SETUP ERROR
004720  104200  002420  001107                                                  MOV     #ER28,HRQ.04
004723  104202  047674                                                          MOV     #28!FTLDEV+4000.,R2
004725  104020  001105                                                          MOV     R2,HRQ.02
004727  104200  004727  001104                                                  MOV     #.,HRQ.01
004732  104200  060014  001103                                                  MOV     #ERRMC,HRQ.RQ
004735  004775                  BR      4$              ; BRANCH TO REPORT
004736  102201  040000          6$:     BIT     #INITW,R1       ; SEE IF INITIAL WRITE
004740  014762                          BEQ     5$              ; IF NOT, BRANCH
004741  103201  040000                  BIC     #INITW,R1       ; NO INITIAL WRITE
004743  100141                          MOV     R1,(R4)         ; SAVE PARAMETERS
004744                                  ASSUME  S.PARM,0
004744                                  MSSG    4               ; REPORT NO INITIAL WRITE
004744  104200  005704  001105                                                  MOV     #MS4,HRQ.02
004747  100467                                                                          MOV R0,-(SP)
004750  104650  000063  001104                                                  MOV     U.UNUM(R5),HRQ.01
004753  105650  000050  001104                                                  ADD     U.SUBU(R5),HRQ.01
004756  104207  060015                                                          MOV     #MESSAG,R0
004760  021053                                                                  CALL    HOSTRQ
004761  104267                                                                          MOV (SP)+,R0
004762          5$:
                ;               BIT     #DCYLS,R1       ; SEE IF IN DIAGNOSTIC AREA
                ;               BEQ     7$              ; IF NOT, BRANCH
                ;               MSSG    5               ; REPORT ASSUMPTION
                ;7$:
004762  104041                  MOV     R4,R1           ; R1 NOW POINTS TO SUBUNIT DATASTRUCTURE
004763  025061                  CALL    CHKBB           ; CHECK BAD BLOCKS
004764  115002                  TST     R2              ; ANY ERRORS ?
004765  054775                  BNE     4$              ; IF SO, BRANCH
004766  104117                  MOV     (R1),R0         ; GET SUBUNIT CHARACTERISTICS
004767                          ASSUME  S.PARM,0
004767  102207  000040          BIT     #BEUSED,R0      ; SEE IF BEGIN/END SETS WERE USED
004771  055023                  BNE     1$              ; IF SO, SKIP EVERYTHING
```

```
     44 004772  025232                              CALL    CHKTG           ; COMPUTE TRACK/GROUP DATABASE
     45 004773  115002                              TST     R2              ; SEE IF ANY ERRORS
     46 004774  015023                              BEQ     1$              ; IF NOT, BRANCH
     47 004775  104650  000063  001106  4$:         MOV     U.UNUM(R5),HRQ.03       ; GET STARTING UNIT NUMBER
     48 005000  105650  000050  001106              ADD     U.SUBU(R5),HRQ.03       ; ADD OFFSET
     49 005003  104307  001103                      MOV     HRQ.RQ,R0       ; SET UP TO REPORT ERROR
     50 005005  021053                              CALL    HOSTRQ          ; REPORT
     51 005006  101200  000002  002223              BIS     #DIE,M.PARM     ; FLAG INITIALIZATION ERROR
     52 005011  104201  000001                      MOV     #U.SUBP,R1      ; R1 WILL POINT AT SUBUNIT POINTERS
     53 005013  105051                              ADD     R5,R1           ; R1 POINTS AT SUBUNIT POINTERS
     54 005014  105651  000050                      ADD     U.SUBU(R5),R1   ; R1 POINTS AT POINTER TO SUBUNIT BEING HANDLED
     55 005016  104117                              MOV     (R1),R0         ; R0 POINTS TO SUBUNIT BEGING HANDLED
     56 005017  104203  100000                      MOV     #DROP,R3        ; MOVE DROP FLAG TO R3
     57 005021  100173                              MOV     R3,(R0)         ; DROP THIS SUBUNIT
     58 005022                                      ASSUME  S.PARM,0
     59 005022  114002                              CLR     R2              ; NO ERRORS
     60 005023  104653  000050          1$:         MOV     U.SUBU(R5),R3   ; GET SUBUNIT OFFSET
     61 005025  115403                              INC     R3              ; NEXT SUBUNIT
     62 005026  100653  000050                      MOV     R3,U.SUBU(R5)   ; SAVE
     63 005030  106203  000003                      CMP     #3,R3           ; SEE IF VALID OFFSET
     64 005032                                      BCS     23$             ; IF NOT, BRANCH
        005032  045034                                                                  BCC     .+2
        005033  005044                                                                  BR      23$
     65 005034  105203  000001                      ADD     #U.SUBP,R3      ; WILL POINT TO NEXT SUBUNIT POINTER
     66 005036  105053                              ADD     R5,R3           ; ADD POINTER TO UNIT DATABASE
     67 005037  104133                              MOV     (R3),R3         ; GET POINTER
     68 005040  075023                              BMI     1$              ; IF NO SUBUNIT, BRANCH
     69 005041  104131                              MOV     (R3),R1         ; GET SUBUNIT PARAMETERS
     70 005042                                      ASSUME  S.PARM,0
     71 005042  075023                              BMI     1$              ; IF DROPPED, BRANCH
     72 005043                                      ASSUME  DROP,100000
     73 005043  005047                              BR      22$             ; EXIT
     74 005044  114003                  23$:        CLR     R3              ; NO SUBUNITS
     75 005045  100653  000050                      MOV     R3,U.SUBU(R5)   ; SAVE
     76 005047  115003                  22$:        TST     R3              ; ANY?
     77 005050  055054                              BNE     2$              ; IF SO, BRANCH
     78 005051  104207  000035                      MOV     #GETSER,R0      ; GETSER NEXT MODULE
     79 005053  005056                              BR      3$              ; EXIT
     80 005054  104207  000033          2$:         MOV     #SCHAR0,R0      ; GO BACK TO FIRST CHARACTERISTICS MODULE
     81 005056  114001                  3$:         CLR     R1              ; IMMIDATE CALL
     82 005057                                      POP     R4              ; RESTORE
        005057  104264                                                                  MOV (SP)+,R4
     83 005060  003231                              BR      JMPRET          ; RETURN TO         R
     84                                             .DSABL  LSB
```

```
                                .SBTTL  CHKBB  - CHECK THE BAD BLOCKS FOR ERRORS
005061                          CHKBB:
                                ;
                                ;       CHECK THE BAD BLOCKS TO ASSURE THEY DON'T EXCEED THE MAXIMUM BLOCK
                                ;       NUMBER
                                ;
005061                                  PUSH    R5              ; SAVE THE REGISTERS
005061  100465                                                                  MOV R5,-(SP)
005062  104613  000012                 MOV     S.BADP(R1),R3   ; GET POINTER TO BAD BLOCKS
005064  015227                         BEQ     5$              ; IF NO BAD BLOCKS, BRANCH
005065  104632  000001          1$:    MOV     1(R3),R2        ; IS THIS THE LAST BAD BLOCK?
005067  075121                         BMI     2$              ; IF SO, BRANCH
005070  104032                         MOV     R3,R2           ; R2 WILL POINT TO NEXT BAD BLOCK
005071  105202  000002                 ADD     #2,R2           ; R2 POINTS TO NEXT BAD BLOCK
005073  024632                         CALL    CBB2            ; COMPARE
005074  045100                         BCC     7$              ; IF ERROR, BRANCH
005075  105203  000002                 ADD     #2,R3           ; POINT TO NEXT BAD BLOCK
005077  005065                         BR      1$              ; LOOP
005100                          7$:    DEVFTL  53,#SER20       ; DUPLICATE BAD BLOCKS
005100  104200  003470  001107                                                  MOV     #ER53,HRQ.04
005103  104200  004052  001110                                                  MOV     #SER20,HRQ.05
005106  104202  047725                                                          MOV     #53!FTLDEV+4000.,R2
005110  104020  001105                                                          MOV     R2,HRQ.02 -
005112  104200  005112  001104                                                  MOV     #.,HRQ.01
005115  104200  060014  001103                                                  MOV     #ERRMC,HRQ.RQ
005120  005230                         BR      6$              ; BRANCH
005121  104117                  2$:    MOV     (R1),R0         ; GET SUBUNIT PARAMETERS
005122                                 ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
005122  102207  020000                 BIT     #DCYLS,R0       ; SEE IF DIAGNOSTIC CYLINDERS IN USE
005124  055173                         BNE     3$              ; IF SO, BRANCH
005125  104612  000007                 MOV     S.SCHR(R1),R2   ; R2 POINTS TO SUBUNIT CHARACTERISTICS
005127  105202  000012                 ADD     #LBNHST,R2      ; R2 POINTS TO NUMBER OF LBNS IN HOST AREA
005131  024632                         CALL    CBB2            ; COMPARE
005132  045134                         BCC     8$              ; IF ERROR, BRANCH
005133  005227                         BR      5$              ; EXIT WITH NO ERROR
005134  104125                  8$:    MOV     (R2),R5         ; GET LO ORDER MAX LBN
005135  107205  000001                 SUB     #1,R5           ; DECREMENT COUNT
005137  100125                         MOV     R5,(R2)         ; SAVE
005140  045146                         BCC     9$              ; IF NO CARRY, BRANCH
005141  104625  000001                 MOV     1(R2),R5        ; GET HI ORDER
005143  117405                         DEC     R5              ; PROPOGATE CARRY
005144  100625  000001                 MOV     R5,1(R2)        ; SAVE
005146                          9$:    DEVFTL  54,<#SER20,(R2)+,(R2)>  ; BAD BLOCK > MAXIMUM
005146  104200  003505  001107                                                  MOV     #ER54,HRQ.04
005151  104200  004052  001110                                                  MOV     #SER20,HRQ.05
005154  104220  001111                                                          MOV     (R2)+,HRQ.06
005156  104120  001112                                                          MOV     (R2),HRQ.07
005160  104202  047726                                                          MOV     #54!FTLDEV+4000.,R2
005162  104020  001105                                                          MOV     R2,HRQ.02
005164  104200  005164  001104                                                  MOV     #.,HRQ.01
005167  104200  060014  001103                                                  MOV     #ERRMC,HRQ.RQ
005172  005230                         BR      6$              ; ERROR EXIT
005173  104032                  3$:    MOV     R3,R2           ; R2 POINTS TO LAS BAD BLOCK
005174  104203  004511                 MOV     #MAXDBN,R3      ; R3 POINTS TO MAXIMUM DBN
005176  024632                         CALL    CBB2            ; COMPARE
005177  045227                         BCC     5$              ; IF NO ERROR, BRANCH
005200                          4$:    DEVFTL  54,<#SER20,MAXDBN,MAXDBN+1> ; BAD BLOCK > MAXIMUM
```

```
UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE 137-1
CHKBB  - CHECK THE BAD BLOCKS FOR ERRORS

        005200  104200  003505  001107                          MOV     #ER54,HRQ.04
        005203  104200  004052  001110                          MOV     #SER20,HRQ.05
        005206  104300  004511  001111                          MOV     MAXDBN,HRQ.06
        005211  104300  004512  001112                          MOV     MAXDBN+1,HRQ.07
        005214  104202  047726                                  MOV     #54!FTLDEV+4000.,R2
        005216  104020  001105                                  MOV     R2,HRQ.02
        005220  104200  005220  001104                          MOV     #.,HRQ.01
        005223  104200  060014  001103                          MOV     #ERRMC,HRQ.RQ
    43  005226  005230                      BR      6$          ; BRANCH TO EXIT
    44  005227  114002              5$:     CLR     R2          ; FLAG AS NO ERRORS
    45  005230                      6$:     POP     R5          ; RESTORE THE REGISTERES
        005230  104265                                                  MOV (SP)+,R5
    46  005231  000000                      RETURN              ; RETURN TO INSCHR
```

```
                                        .SBTTL  CHKTG  - CHECK THE TRACK/GROUPS FOR ERRORS, AND CONVERT TO BN'S
005232                                  CHKTG:
                                        ;
                                        ;       CONVERT THE TRACKS/GROUPS TO SECTOR COUNTS, AND FIND THE MAXIMUM
                                        ;       NUMBER OF 'LOOPS' THAT SETUP CAN RUN THROUGH TO TEST ALL
                                        ;       TRACKS ON THE TESTED CYLINDERS ONLY
                                        ;
                                        ;
                                        ;       FIRST COPY THE TRACKS/GROUPS TO A TEMP LOCATION
                                        ;
005232                                          PUSH    R5                      ; SAVE R5
005232  100465                                                                  MOV R5,-(SP)
005233  104207  004516                         MOV     #TGS,R0                 ; POINT TO TEMPORARY STORAGE LOCATION
005235  104202  000017                         MOV     #S.TGSS,R2              ; R2 WILL POINT TO T/G LIST
005237  105012                                 ADD     R1,R2                   ; R2 POINTS TO T/G LIST
005240  104225                         1$:     MOV     (R2)+,R5                ; GET WORD
005241  100275                                 MOV     R5,(R0)+                ; SAVE IN TEMP LOCATION
005242  035240                                 BPL     1$                      ; COPY ENTIRE LIST
                                        ;
                                        ;       NOW MOVE THE STARTING CYLINDER TO THE INITIAL OFFSET AREA
                                        ;
005243  104612  000016                         MOV     S.TGOF+1(R1),R2         ; GET HI ORDER STARTING CYL
005245  103202  170000                         BIC     #^CHBHINB,R2            ; CLEAR UNUSED BITS
005247  100612  000016                         MOV     R2,S.TGOF+1(R1)         ; SAVE
                                        ;
                                        ;       CHECK THAT THE MAXIMUM TRACK/GROUP IS VALID
                                        ;
005251  104117                                 MOV     (R1),R0                 ; GET SUBUNIT PARAMETERS
005252                                         ASSUME  S.PARM,0                ; ASSUME THAT S.PARM IS ZERO
005252  104642  000007                         MOV     S.SCHR(R4),R2           ; R2 POINTS TO SUBUNIT CHAR
005254  102207  000020                         BIT     #TRACKS,R0              ; SEE IF PROCESSING GROUPS OR TRACKS
005256  055265                                 BNE     2$                      ; IF TRACKS, BRANCH
005257  104627  000002                         MOV     GRPCYL(R2),R0           ; GET GROUPS PER CYLINDER
005261  104200  005516  005514                 MOV     #GRP$,TG$               ; PRINT 'GROUP' IF ERROR
005264  005272                                 BR      3$                      ; BRANCH
005265  104627  000003                 2$:     MOV     TRKGRP(R2),R0           ; GET TRACKS PER GROUP
005267  104200  005512  005514                 MOV     #TRK$,TG$               ; PRINT 'TRACK' IF ERROR
005272  103207  177400                 3$:     BIC     #^HIBYTE,R0             ; CLEAR UNUSED BITS
005274  104202  004516                         MOV     #TGS,R2                 ; POINT TO TRACK/GROUP LIST
005276  104223                         4$:     MOV     (R2)+,R3                ; SEE IF THIS WORD IS END-OF-LIST
005277  035276                                 BPL     4$                      ; IF NOT, BRANCH
005300  103203  177400                         BIC     #HIBYTE,R3              ; CLEAR EOL FLAG
005302  106037                                 CMP     R3,R0                   ; SEE IF WITHIN LIMITS
005303  075333                                 BMI     5$                      ; IF SO, BRANCH
005304  117407                                 DEC     R0                      ; ADJUST T/G MAX FOR 0-N
005305                                         DEVFTL  58,<#SER20,TG$,R0>      ; REPORT ERROR
005305  104200  003722  001107                                                 MOV     #ER58,HRQ.04
005310  104200  004052  001110                                                 MOV     #SER20,HRQ.05
005313  104300  005514  001111                                                 MOV     TG$,HRQ.06
005316  104070  001112                                                         MOV     R0,HRQ.07
005320  104202  047732                                                         MOV     #58!FTLDEV+4000.,R2
005322  104020  001105                                                         MOV     R2,HRQ.02
005324  104200  005324  001104                                                 MOV     #.,HRQ.01
005327  104200  060014  001103                                                 MOV     #ERRMC,HRQ.RQ
005332  005512                                 BR      17$                     ; EXIT
```

```
                                        ;
                                        ;       NOW COMPUTE THE NUMBER OF SECTORS FROM THE START OF THE CYLINDER/GROUP
                                        ;       TO THE FIRST TRACK/GROUP TO TEST
                                        ;
005333  104207  004516         5$:     MOV     #TGS,R0                 ; R0 POINTS TO TRACK/GROUP LIST
005335  104203  000015                 MOV     #S.TGOF,R3              ; R3 WILL POINT TO T/G INITIAL OFFSET
005337  105013                         ADD     R1,R3                   ; R3 POINTS TO T/G INITIAL OFFSET
005340  104172                         MOV     (R0),R2                 ; GET FIRST T/G
005341  103202  177400                 BIC     #HIBYTE,R2              ; CLEAR UNUSED BITS
005343  025515                         CALL    COMPDP                  ; COMPUTE SECTORS FROM CYL TO INIT T/G
005344  105135                         ADD     (R3),R5                 ; ADD TO INITIAL OFFSET
005345  100235                         MOV     R5,(R3)+                ; MOVE BACK
005346  045352                         BCC     6$                      ; IF NO CARRY, BRANCH
005347  104135                         MOV     (R3),R5                 ; GET HI ORDER WORD
005350  115405                         INC     R5                      ; PROPOGATE CARRY
005351  100135                         MOV     R5,(R3)                 ; SAVE
005352  115403                 6$:     INC     R3                      ; POINT TO NEXT AREA
                                        ;
                                        ;       NOW COMPUE HOW MANY SECTORS FROM THE LAST TRACK/GROUP TO TEST TO THE
                                        ;       NEXT TRACK/GROUP TO TEST
                                        ;
005353  104174                 7$:     MOV     (R0),R4                 ; GET TRACK/GROUP
005354  075413                         BMI     8$                      ; IF EOL, BRANCH
005355  104672  000001                 MOV     1(R0),R2                ; GET NEXT TRACK
005357  103202  177400                 BIC     #HIBYTE,R2              ; CLEAR EOL FLAG, IF ANY
005361  107272                         SUB     (R0)+,R2                ; GET HOW MANY T/G'S BETWEEN LAST/NEXT
005362  015364                         BEQ     20$                     ; IF ZERO, ERROR
005363  035410                         BPL     18$                     ; SHOULD BE AT LEAST 1 T/G BETWEEN
005364                          20$:    DEVFTL  59,<#SER20,TG$>         ; REPORT ERROR
005364  104200  003753  001107                         MOV     #ER59,HRQ.04
005367  104200  004052  001110                         MOV     #SER20,HRQ.05
005372  104300  005514  001111                         MOV     TG$,HRQ.06
005375  104202  047733                                 MOV     #59!FTLDEV+4000.,R2
005377  104020  001105                                 MOV     R2,HRQ.02
005401  104200  005401  001104                         MOV     #.,HRQ.01
005404  104200  060014  001103                         MOV     #ERRMC,HRQ.RQ
005407  005512                         BR      17$                     ; EXIT
005410  025515                 18$:    CALL    COMPDP                  ; COMPUTE HOW MANY SECTORS TO NEXT T/G
005411  100235                         MOV     R5,(R3)+                ; STORE THE OFFSET
005412  005353                         BR      7$                      ; LOOP
                                        ;
                                        ;       NOW COMPUTE HOW MANY SECTORS FROM THE LAST T/G TO TEST TO THE FIRST
                                        ;
005413  104115                 8$:     MOV     (R1),R5                 ; GET SUBUNIT PARAMETERS
005414                                  ASSUME  S.PARM,0                ; ASSUME THAT S.PARM IS ZERO
005414  104612  000007                 MOV     S.SCHR(R1),R2           ; R2 POINTS TO SUBUNIT CHAR
005416  102205  000020                 BIT     #TRACKS,R5              ; SEE IF USING GROUPS OR TRACKS
005420  055424                         BNE     9$                      ; IF TRACKS, BRANCH
005421  104622  000002                 MOV     GRPCYL(R2),R2           ; GET GROUPS/CYL
005423  005426                         BR      10$                     ; BRANCH
005424  104622  000003          9$:    MOV     TRKGRP(R2),R2           ; GET TRACKS/GROUP
005426  105302  004516          10$:   ADD     TGS,R2                  ; ADD STARTING T/G
005430  103202  177400                 BIC     #HIBYTE,R2              ; CLEAR UNUSED BITS
005432  103204  177400                 BIC     #HIBYTE,R4              ; CLEAR UNUSED BITS
005434  107042                         SUB     R4,R2                   ; FIND HOW MANY T/G'S TO SKIP
005435  025515                         CALL    COMPDP                  ; COMPUTE HOW MANY SECTORS
005436  100235                         MOV     R5,(R3)+                ; STORE IN LIST
```

```
                                        CLR     R5                      ; SET UP FOR MARKING EOL
                                        MOV     R5,(R3)+                ; STORE EOL
                                ;
                                ;       NOW SEE HOW MANY TIMES THE SETUP ROUTINE CAN GO THROUGH THE LIST
                                ;       WITHOUT EXCEEDING THE MAXIMUM SECTOR NUMBER
                                ;
                                        MOV     #S.BESS,R0              ; R0 WILL POINT TO THE MAX SEC NUMBER
                                        ADD     R1,R0                   ; R0 POINTS TO THE MAX SECTOR NUMBER
                                        MOV     S.TGOF-S.BESS(R0),R4    ; R4 HAS LO ORDER INITIAL OFFSET
                                        MOV     S.TGOF-S.BESS+1(R0),R5  ; R5 HAS HI ORDER INITIAL OFFSET
                                        PUSH    R1                      ; SAVE R1
005450  100461                                                                  MOV R1,-(SP)
                                        CLR     R2                      ; CLEAR LO ORDER MAX COUNT
                                        CLR     R3                      ; CLEAR HI ORDER MAX COUNT
                                        MOV     #S.TGSS-S.BESS,R1       ; R1 WILL POINT TO SECTOR OFFSET AREA
                                        ADD     R0,R1                   ; R1 POINTS TO SECTOR OFFSET AREA
                                13$:    MOV     R4,-(SP)                ; SAVE LO ORDER TOTAL ON STACK
                                        MOV     (R1)+,R4                ; GET LO ORDER SECTOR OFFSET
                                        BNE     14$                     ; IF NOT EOL, BRANCH
                                        MOV     R0,R1                   ; R1 WILL POINT TO START OF OFFSET LIST
                                        ADD     #S.TGSS-S.BESS,R1       ; R1 POINTS TO START OF OFFSET LIST
                                        MOV     (R1)+,R4                ; GET NEW OFFSET
                                14$:    ADD     (SP)+,R4                ; ADD TO LOW ORDER SECTOR TOTAL
                                        BCC     15$                     ; IF NO CARRY, BRANCH
                                        INC     R5                      ; PROPOGATE CARRY
                                15$:    CMP     1(R0),R5                ; SEE IF MAX EXCEEDED
                                        BCS     16$                     ; IF SO, BRANCH
005472  045474                                                                 BCC     .+2
005473  005505                                                                 BR      16$
                                        BNE     11$                     ; IF YOUR SURE IT'S OK, BRANCH
                                        CMP     (R0),R4                 ; SEE IF MAX EXCEEDED
                                        BCS     16$                     ; IF SO, BRANCH
005476  045500                                                                 BCC     .+2
005477  005505                                                                 BR      16$
                                11$:    ADD     #1,R2                   ; ADD 1 TO COUNT
                                        BCC     13$                     ; IF NO CARRY, BRANCH
                                        INC     R3                      ; PROPOGATE CARRY
                                        BR      13$                     ; BRANCH
                                16$:    POP     R1                      ; RESTORE R1
005505  104261                                                                 MOV (SP)+,R1
                                        MOV     R2,(R0)                 ; STORE LOW ORDER MAXIMUM COUNT
                                        MOV     R3,1(R0)                ; STORE HI MAXIMUM COUNT
                                        CLR     R2                      ; CLEAR R2
                                17$:    POP     R5                      ; RESTORE
005512  104265                                                                 MOV (SP)+,R5
                                        RETURN

                                TG$:    .BLKW   1
```

COMPDP - CALCULATE SECTORS/TRACKS OR SECTORS/GROUPS

```
                                        .SBTTL  COMPDP - CALCULATE SECTORS/TRACKS OR SECTORS/GROUPS
005515                                  COMPDP:
                                        ;
                                        ;       COMPUTE HOW MANY SECTORS ARE IN THE NUMBER OF TRACKS OR GROUPS
                                        ;       PASSED IN R2
                                        ;
005515  114005                                  CLR     R5                      ; CLEAR RUNNING TOTAL
005516  104114                                  MOV     (R1),R4                 ; GET SUBUNIT PARAMETERS
005517                                          ASSUME  S.PARM,0                ; ASSUME THAT S.PARM IS ZERO
005517  005530                                  BR      3$                      ; SEE IF IMMIDATE EXIT
005520  102204  000020                  1$:     BIT     #TRACKS,R4              ; SEE IF USING GROUPS OR TRACKS
005522  055526                                  BNE     2$                      ; IF TRACKS, BRANCH
005523  105305  004514                          ADD     SECGRP,R5               ; ADD SECTORS/GROUP TO RUNNING TOTAL
005525  005530                                  BR      3$                      ; BRANCH
005526  105305  004513                  2$:     ADD     SECTRK,R5               ; ADD SECTORS/TRACK TO RUNNING TOTAL
005530  117402                          3$:     DEC     R2                      ; DECREMENT COUNT
005531  035520                                  BPL     1$                      ; IF INCOMPLETE, BRANCH
005532  000000                                  RETURN                          ; RETURN TO CALLING PROGRAM
                                                .IF     LE,MAXADR-.
                                        MAXADR  =       .+1
                                                .ENDC
```

```
                                        .SBTTL  ********** OVERLAY MODULE GETSER - GET THE HDA AND DRIVE SERIAL NUMBER
                                                DMOVLY  GS,AREAI
005533  000105                                  .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;
                                        ;
000035                                  GETSER  =       SCHAR1+1
                                        ;
                                        ;       GET HDA AND DRIVE SERIAL NUMBER (IF POSSIBLE) AND RETURN TO HOST
                                        ;
                                                .ENABL  LSB
004710                                          PUSH    R4                      ; SAVE SUBUNIT POINTER
004710  100464                                                                  MOV R4,-(SP)
004711  104650  000063  005335                 MOV     U.UNUM(R5),SUNIT        ; SAVE UNIT NUMBER
004714  104651  000063                         MOV     U.UNUM(R5),R1           ; R1 HAS UNIT NUMBER
004716  105201  000003                         ADD     #3,R1                   ; R1 HAS LAST UNIT NUMBER
004720  104207  000001                         MOV     #U.SUBP,R0              ; R0 WILL POINT TO SUBUNIT POINTERS
004722  105057                                 ADD     R5,R0                   ; R0 POINTS TO SUBUNIT POINTERS
004723  104274                          11$:   MOV     (R0)+,R4                ; R4 POINTS TO SUBUNIT
004724  075321                                 BMI     12$                     ; IF NO SUBUNIT, BRANCH
004725  104203  001750                         MOV     #1000.,R3               ; INITILIZE SEEK COUNTER
004727  100643  000001                         MOV     R3,S.SEEK(R4)           ; SAVE SEEK COUNTER
004731                                         PUSH    <R0,R1>                 ; SAVE POINTER AND COUNT
004731  100467                                                                  MOV R0,-(SP)
004732  100461                                                                  MOV R1,-(SP)
                                        .SBTTL  REPSER - FIND AND REPORT DRIVE AND HDA SERIAL NUMBER
                                        ;REPSER
                                        ;
                                        ;       FIND AND REPORT DRIVE AND HDA SERIAL NUMBER.  IF ERROR OCCURRS,
                                        ;       REPORT SERIAL NUMBER AS ZERO.  NO ERROR IS RETURNED FROM THIS ROUTINE
                                        ;
004733  021777                                 CALL    BULDUM                  ; BUILD DUMMY SDI BLOCK
004734  104641  000007                         MOV     S.SCHR(R4),R1           ; R1 POINTS TO SUBUNIT CHARACTERISTICS
004736  104610  000000  002173                 MOV     LBNCYL(R1),STACYL       ; MOVE LO STARTING CYL
004741  104610  000001  002174                 MOV     LBNCYL+1(R1),STACYL+1   ; MOVE HI STARTING CYL
004744  104617  000004                         MOV     RBNTRK(R1),R0           ; GET RBNS/TRACK
004746  103207  177600                         BIC     #HIBYTE!200,R0          ; CLEAR UNUSED BITS
004750  105617  000011                         ADD     LBNTRK(R1),R0           ; ADD LBNS/TRACK
004752  103207  177400                         BIC     #HIBYTE,R0              ; CLEAR UNUSED BITS
004754  104070  002177                         MOV     R0,LRDTRK               ; MOVE TO SECTORS PER TRACK
004756  114000  002200                         CLR     RBNFLG                  ; NO RBN FLAG
004760  104657  000057                         MOV     U.COPY(R5),R0           ; GET NUMBER OF XBN COPIES
004762  105077                                 ADD     R0,R0                   ; DOUBLE
004763  104070  005336                         MOV     R0,SERRTY               ; RETRY 2 TIMES THE NUMBER OF COPIES
004765  114007                          1$:    CLR     R0                      ; TO SET UP COPIES AND SETUP CALC
004766  100657  000060                         MOV     R0,U.CCOP(R5)           ; SAVE
004770  104641  000007                         MOV     S.SCHR(R4),R1           ; GET POINTER TO SUBUNIT CHARACTERISTICS
004772  114000  002175                         CLR     CURBN                   ; PUT LO ORDER XBN IN CURBN
004774  114000  002176                         CLR     CURBN+1                 ; PUT HI ORDER XBN IN CURBN+1
004776  104207  002173                  2$:    MOV     #CALCSC,R0              ; POINT TO CALCULATION AREA
005000  104641  000007                         MOV     S.SCHR(R4),R1           ; POINT TO SUBUNIT CHARACTERISTICS
005002  060020                                 XFC     CVT                     ; CONVERT
005003  023720                                 CALL    RECOVR                  ; CLEAR ALL POSSIBLE ERRORS
```

```
 54 005004  115002                          TST     R2                      ; SEE IF ERROR OCCURRED
 55 005005  055207                          BNE     8$                      ; IF SO, BRANCH
 56 005006  104203  001655                  MOV     #CR.INR,R3              ; SET UP FOR RECALIBRATE
 57 005010  021173                          CALL    TALK                    ; RECALIBRATE DRIVE
 58 005011  115002                          TST     R2                      ; SEE IF ERROR OCCURRED
 59 005012  055207                          BNE     8$                      ; IF SO, BRANCH
 60 005013  104300  002201  001673          MOV     CYL,LOCYL               ; MOVE LO CYL TO SEEK COMMAND
 61 005016  104300  002202  001674          MOV     CYL+1,LOCYL+1           ; MOVE TO SEEK COMMAND
 62 005021  104300  002203  001675          MOV     GROUP,LOCYL+2           ; MOVE GROUP TO SEEK COMMAND
 63 005024  104203  001636                  MOV     #CR.SEK,R3              ; POINT TO SEEK COMMAND
 64 005026  021173                          CALL    TALK                    ; SEND SEEK
 65 005027  115002                          TST     R2                      ; SEE IF ERROR OCCURRED
 66 005030  055207                          BNE     8$                      ; IF SO, BRANCH
 67 005031  104651  000011                  MOV     U.MSTO(R5),R1           ; GET MASTER SEEK TIMEOUT
 68 005033  100651  000005                  MOV     R1,U.TIMH(R5)           ; SAVE
 69 005035  114001                          CLR     R1                      ; FOR LOW ORDER TIMEOUT
 70 005036  100651  000006                  MOV     R1,U.TIML(R5)
 71 005040  104201  000310          3$:     MOV     #200.,R1                ; INNER LOOP TIMEOUT
 72 005042  117401                  4$:     DEC     R1                      ; DECREMENT INNER COUNTER
 73 005043  055042                          BNE     4$                      ; IF UNEXPIRED, BRANCH
 74 005044  020775                          CALL    RTDSL                   ; GET REAL TIME DRIVE STATE
 75 005045  115002                          TST     R2                      ; SEE IF ERROR OCCURRED
 76 005046  055207                          BNE     8$                      ; IF SO, BRANCH
 77 005047  115001                          TST     R1                      ; SEE IF READ/WRITE READY IS ASSERTED
 78 005050                                  ASSUME  RWRDY,100000
 79 005050  075070                          BMI     5$                      ; IF SO, BRANCH
 80 005051  104651  000006                  MOV     U.TIML(R5),R1           ; GET TIMEOUT
 81 005053  107201  000001                  SUB     #1,R1                   ; DECREMENT TIMEOUT
 82 005055  100651  000006                  MOV     R1,U.TIML(R5)           ; SAVE
 83 005057  045040                          BCC     3$                      ; LOOP IF INCOMPLETE
 84 005060  104651  000005                  MOV     U.TIMH(R5),R1           ; GET HI ORDER TIMEOUT
 85 005062  107201  000001                  SUB     #1,R1                   ; DECREMENT TIMEOUT
 86 005064  100651  000005                  MOV     R1,U.TIMH(R5)           ; SAVE
 87 005066  045040                          BCC     3$                      ; IF TIMOUT UNEXPIRED, BRANCH
 88 005067  005207                          BR      8$                      ; ERROR
 89 005070  104200  100000  005337  5$:     MOV     #RSTOP,SERCHN+RW.STAT   ; MOVE LAST CHAIN FLAG TO CHAIN
 90 005073  104300  002175  005341          MOV     CURBN,SERCHN+RW.LOW     ; MOVE TO OUTPUT
 91 005076  104641  000007                  MOV     S.SCHR(R4),R1           ; R1 POINTS TO SUBUNIT CHARACTERISTICS
 92 005100  104611  000002                  MOV     HIXBN(R1),R1            ; GET HI XBN BITS
 93 005102  110601                          ROR     R1                      ; ROTATE TO CORRECT POSITION
 94 005103  110601                          ROR     R1                      ; ROTATE TO CORRECT POSITION
 95 005104  110601                          ROR     R1                      ; ROTATE TO CORRECT POSITION
 96 005105  110601                          ROR     R1                      ; ROTATE TO CORRECT POSITION
 97 005106  103201  170377                  BIC     #HBLONB,R1              ; CLEAR UNUSED BITS
 98 005110  101201  120000                  BIS     #HD.XBN,R1              ; MAKE A XBN
 99 005112  105301  002176                  ADD     CURBN+1,R1              ; SET IN HI XBN BITS
100 005114  104010  005342                  MOV     R1,SERCHN+RW.HI         ; SAVE
101 005116  104300  002204  005343          MOV     TRACK,SERCHN+RW.CMD     ; MOVE TRACK TO CHAIN
102 005121  101200  013400  005343          BIS     #RREAL,SERCHN+RW.CMD    ; SET IN REAL TIME COMMAND
103 005124  104200  002207  005344          MOV     #DUMSDI,SERCHN+RW.SDI   ; POINT TO DUMMY SDI BLOCK
104 005127  104652  000025                  MOV     U.MASK(R5),R2           ; R2 HAS UDA PORT MASK
105 005131  104207  005337                  MOV     #SERCHN,R0              ; R0 POINTS TO CHAIN
106 005133  060012                          XFC     WAITSI                  ; WAIT FOR SECTOR OR INDEX
107 005134  115001                          TST     R1                      ; SEE IF ERROR OCCURRED
108 005135  055207                          BNE     8$                      ; IF SO, BRANCH
109 005136  060002                          XFC     XREAD                   ; READ THE SECTOR
110 005137  106201  000004                  CMP     #4,R1                   ; SEE IF XBN HEADER COMPARE FAILURE
```

```
                                                BEQ     7$                      ; IF SO, TRY NEXT COPY
                                                TST     R1                      ; SEE IF AN ERROR OCCURRED
                                                BNE     8$                      ; IF SO, BRANCH
005144  102200  010000  005337                  BIT     #ECCFLG,SERCHN+RW.STAT  ; SEE IF BUFFER HAS AN ECC ERROR
005147  015160                                  BEQ     6$                      ; IF NOT, BRANCH
005150  104207  005337                          MOV     #SERCHN,R0              ; POINT TO CHAIN
005152  060015                                  XFC     ECC                     ; CORRECT THE BUFFER
005153  115001                                  TST     R1                      ; SEE IF ERROR
005154  055166                                  BNE     7$                      ; IF SO, BRANCH
005155  106657  000032                          CMP     U.ECCT(R5),R0           ; SEE IF CORRECTIONS EXCEED THRESHOLD
                                        ;
                                        ;       NOTE: I ALLOW 1 ERROR MORE THAN SPEC'D ECC THRESHOLD
                                        ;
005157  075166                          6$:     BMI     7$                      ; IF SO, BRANCH
005160  104207  005345                          MOV     #SERSEC,R0              ; R0 POINTS TO BUFFER
005162  021146                                  CALL    CMPEDC                  ; COMPUTE EDC OVER BUFFER
005163  106020  005745                          CMP     R2,SERSEC+BF.EDC        ; SEE IF EDC'S MATCH
005165  015223                                  BEQ     9$                      ; IF SO, YOU'VE GO IT!!!!!!!!!!!!!
005166  104651  000060                  7$:     MOV     U.CCOP(R5),R1           ; GET CURRENT COPY NUMBER
005170  115401                                  INC     R1                      ; INCREMENT COUNT
005171  100651  000060                          MOV     R1,U.CCOP(R5)           ; SAVE
005173  106651  000057                          CMP     U.COPY(R5),R1           ; SEE IF ALL COPIES TRIED
005175  015207                                  BEQ     8$                      ; IF SO, BRANCH
005176  104647  000007                          MOV     S.SCHR(R4),R0           ; R1 POINTS TO SUBUNIT CHARACTERISTICS
005200  105670  000010  002175                  ADD     FCTSIZ(R0),CURBN        ; ADD TO CURRENT BN
005203  044776                                  BCC     2$                      ; CALCULATE AND TRY NEXT SECTOR
005204  115400  002176                          INC     CURBN+1                 ; PROPOGATE CARRY
005206  004776                                  BR      2$                      ; BRANCH
005207  117400  005336                  8$:     DEC     SERRTY                  ; DECREMENT RETRY COUNT
005211  054765                                  BNE     1$                      ; IF UNEXHAUSTED, BRANCH
                                        ;
                                        ;       COULDN'T GET THE SERIAL NUMBER -- REPORT 0'S AND 512 DRIVE
                                        ;
005212  114000  005347                          CLR     SERSEC+2                ; ZERO HDA SER #
005214  114000  005350                          CLR     SERSEC+3                ; ZERO HDA SER #
005216  114000  005351                          CLR     SERSEC+4                ; ZERO HDA SER #
005220  114000  005352                          CLR     SERSEC+5                ; ZERO HDA SER #
005222  005264                                  BR      10$                     ; ASSUME 512 DRIVE
005223  106200  126736  005345  9$:             CMP     #MOD512,SERSEC          ; SEE IF 512 BYTE MODE DRIVE
005226  015264                                  BEQ     10$                     ; IF SO, BRANCH
005227  104147                                  MOV     (R4),R0                 ; GET UNIT PARAMETERS
005230                                          ASSUME  S.PARM,0
005230  102207  020000                          BIT     #DCYLS,R0               ; SEE IF USING THE DIAGNOSTIC AREA
005232  055264                                  BNE     10$                     ; IF SO, OK TO TEST, BRANCH
005233                                          DEVFTL  21,SERSEC               ; SET UP REPORT
005233  104200  002111  001107                                                          MOV     #ER21,HRQ.04
005236  104300  005345  001110                                                          MOV     SERSEC,HRQ.05
005241  104202  047665                                                                  MOV     #21!FTLDEV+4000.,R2
005243  104020  001105                                                                  MOV     R2,HRQ.02
005245  104200  005245  001104                                                          MOV     #.,HRQ.01
005250  104200  060014  001103                                                          MOV     #ERRMC,HRQ.RQ
005253  104300  005335  001106                  MOV     SUNIT,HRQ.03            ; MOVE UNIT NUMBER TO MESSAGE
005256  104307  001103                          MOV     HRQ.RQ,R0               ; GET REQUEST NUMBER
005260  021053                                  CALL    HOSTRQ                  ; REPORT TO HOST
005261  101200  000002  002223                  BIS     #DIE,M.PARM             ; DO NOT TEST
005264  104300  005335  001104  10$:             MOV     SUNIT,HRQ.01            ; MOVE IN UNIT NUMBER
005267  104300  004525  001105                  MOV     DSERNM,HRQ.02           ; MOVE DRIVE SERIAL NUMBER TO BUFFER
```

```
    162 005272  104300  004526  001106          MOV     DSERNM+1,HRQ.03         ; MOVE DRIVE SERIAL NUMBER TO BUFFER
    163 005275  104300  004527  001107          MOV     DSERNM+2,HRQ.04         ; MOVE DRIVE SERIAL NUMBER TO BUFFER
    164 005300  104300  005347  001110          MOV     SERSEC+2,HRQ.05         ; MOVE HDA SERIAL NUMBER TO BUFFER
    165 005303  104300  005350  001111          MOV     SERSEC+3,HRQ.06         ; MOVE HDA SERIAL NUMBER TO BUFFER
    166 005306  104300  005351  001112          MOV     SERSEC+4,HRQ.07         ; MOVE HDA SERIAL NUMBER TO BUFFER
    167 005311  104300  005352  001113          MOV     SERSEC+5,HRQ.08         ; MOVE HDA SERIAL NUMBER TO BUFFER
    168 005314  104207  060004                  MOV     #T4UPRM,R0              ; MOVE IN REQUEST
    169 005316  021053                          CALL    HOSTRQ                  ; REPORT
    170 005317                                  POP     <R1,R0>                 ; RESTORE POINTER AND COUNT
        005317  104261                                                                      MOV (SP)+,R1
        005320  104267                                                                      MOV (SP)+,R0
    171 005321  115400  005335          12$:    INC     SUNIT                   ; INCREMENT ACTIVE SUBUNIT
    172 005323  106010  005335                  CMP     R1,SUNIT                ; SEE IF ALL SUBUNITS REPORTED
    173 005325  044723                          BCC     11$                     ; IF NOT, BRANCH
    174 005326                                  POP     R4                      ; RESTORE SUBUNIT POINTER
        005326  104264                                                                      MOV (SP)+,R4
    175 005327  114002                          CLR     R2                      ; NO ERRORS
    176 005330  114001                          CLR     R1                      ; IMMIDATE CALL
    177 005331  104207  000036                  MOV     #INITD,R0               ; INITD NEXT ROUTINE
    178 005333  024212                          CALL    DSABLE                  ; DISABLE ERROR RECOVERY
    179 005334  003231                          BR      JMPRET                  ; RETURN TO         R
    180                                         .DSABL  LSB
```

ASSOCIATED VARIABLES FOR GETTING THE 4DA SERIAL NUMBER

```
      1                                 .SBTTL  ASSOCIATED VARIABLES FOR GETTING THE HDA SERIAL NUMBER
      2 005335  000000          SUNIT:  .WORD   0                       ; THE ACTIVE UNIT NUMBER
      3 005336  000000          SERRTY: .WORD   0                       ; RETRY COUNT
      4 005337  00000C          SERCHN: .WORD   0                       ; THE CHAIN FOR READING SECTOR 0 (RCT)
      5 005340  005345                  .WORD   SERSEC                  ; POINTER TO THE SECTOR
      6 005341  000000                  .WORD   0
      7 005342  000000                  .WORD   0
      8 005343  000000                  .WORD   0
      9 005344  000000                  .WORD   0
     10 005345                  SERSEC:                                 ; 269 WORD BUFFER FOR READ
     11                                 .IF     LE,MAXADR-<.+269.>
     12                         MAXADR  =       .+269.
     13                                 .ENDC
```

```
                                        .SBTTL  ********** OVERLAY MODULE INITD - INITILIZE THE DRIVE  PARAMETERS FOR TESTING
005345                                          DMOVLY  ID,AREAI
005345  000105                                  .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;*******************************************************************************
                                        ;
                                        ;
000036                                  INITD   =       GETSER+1
                                        ;
                                        ;       BRING DRIVE ONLINE, CLEAR ALL ERRORS (IF ANY) AND CHANGE THE
                                        ;       MODE TO THE REQUIRED VALUES
                                        ;
                                        .ENABL  LSB
004710  104207  000004                        MOV     #4,R0           ; MAXIMUM OF 4 SUBUNITS
004712  104201  000001                        MOV     #U.SUBP,R1      ; R1 WILL POINT TO SUBUNIT POINTERS
004714  105051                                ADD     R5,R1           ; R1 POINTS TO SUBUNIT POINTERS
004715  104212                          1$:   MOV     (R1)+,R2        ; GET POINTER TO SUBUNIT INFO
004716  074734                                BMI     2$              ; IF UNIT NOT TESTED, BRANCH
004717  104123                                MOV     (R2),R3         ; GET SUBUNIT PARAMETERS
004720                                        ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
004720  102203  000100                        BIT     #       3       ; SEE IF        AL SEEKS
004722  054726                                BNE     7$              ; IF SO, BRANCH
004723  114000  002217                        CLR     SCR1            ; FLAG AS RANDOM LBNS
004725  004731                                BR      8$              ; BRANCH
004726  104200  177777  002217        7$:   MOV     #-1,SCR1        ; FLAG AS         AL LBNS
004731  102203  040000                  8$:   BIT     #INITW,R3       ; SEE IF SUBUNIT WILL BE INITALLY WRITTEN
004733  054737                                BNE     3$              ; IF SO, BRANCH
004734  117407                          2$:   DEC     R0              ; DECREMENT COUNT
004735  054715                                BNE     1$              ; IF NOT ALL SUBUNITS CHECKED, BRANCH
004736  004750                                BR      4$              ; BRANCH
004737  104657  000046                  3$:   MOV     U.PARM(R5),R0   ; GET UNIT PARAMETERS
004741  101207  040000                        BIS     #INITW,R0       ; FLAG UNIT AS SUBUNITS GET INITIALLY WRITTEN
004743  100657  000046                        MOV     R0,U.PARM(R5)   ; SAVE
004745  101200  100000  002223                BIS     #IWIPRG,M.PARM  ; FLAG MASTER PARAMETERS AS INITIAL WRITE TO BE DONE
004750  104207  000004                  4$:   MOV     #4,R0           ; MAXIMUM OF FOUR SUBUNITS
004752  104201  000005                        MOV     #U.SUBP+4,R1    ; R1 WILL POINT TO AFTERS SUBUNIT POINTERS
004754  105051                                ADD     R5,R1           ; R1 POINTS TO AFTERS SUBUNIT POINTERS
004755  114003                                CLR     R3              ; INITILIZE SUBUNIT PROTECTION FLAGS
004756  110203                          5$:   ROL     R3              ; ROTATE R3
004757  103203  000001                        BIC     #1,R3           ; CLEAR LO BIT
004761  104412                                MOV     -(R1),R2        ; GET SUBUNIT POINTER
004762  075031                                BMI     6$              ; IF NO SUBUNIT, BRANCH
004763  104122                                MOV     (R2),R2         ; GET SUBUNIT STATUS
004764                                        ASSUME  S.PARM,0        ; ASSUME THAT S.PARM IS ZERO
004764  102202  000100                        BIT     #       2       ; SEE IF        AL SEEKS
004766  054773                                BNE     9$              ; IF SO, BRANCH
004767  115000  002217                        TST     SCR1            ; SEE IF ALL SUBUNITS RANDOM SEEKS
004771  054776                                BNE     10$             ; IF NOT, BRANCH
004772  005024                                BR      11$             ; BRANCH
004773  115000  002217                  9$:   TST     SCR1            ; SEE IF ALL SUBUNITS         AL SEEKS
004775  055024                                BNE     11$             ; IF SO, BRANCH
004776                                  10$:  DEVFTL  56,#SER20       ; REPORT ERROR
004776  104200  003607  001107                                                MOV     #ER56,HRQ.04
005001  104200  004052  001110                                                MOV     #SER20,HRQ.05
```

```
        005004  104202  047730                                                  MOV     #56!FTLDEV+4000.,R2
        005006  104020  001105                                                  MOV     R2,HRQ.02
        005010  104200  005010  001104                                         MOV     #.,HRQ.01
        005013  104200  060014  001103                                         MOV     #ERRMC,HRQ.RQ
     55 005016                                  ENDERR  0
        005016  114000  002230                                                  CLR     ERRPOS          ; CLEAR THE POSITION
     56 005020  101200  000002  002223          BIS     #DIE,M.PARM     ; FLAG INITIALIZATION ERROR
     57 005023  005047                          BR      DRVEXT          ; EXIT
     58 005024  102202  004000          11$:    BIT     #RONLY,R2       ; SEE IF READ ONLY DRIVE
     59 005026  015031                          BEQ     6$              ; IF NOT READ ONLY, BRANCH
     60 005027  101203  000001                  BIS     #1,R3           ; SET READ ONLY BIT
     61 005031  117407                  6$:     DEC     R0              ; DECREMENT COUNT
     62 005032  054756                          BNE     5$              ; IF COUNT UNEXPIRED, BRANCH
     63 005033  110203                          ROL     R3              ; ROTATE MASK TO CORRECT POSITION
     64 005034  110203                          ROL     R3              ; ROTATE MASK TO CORRECT POSITION
     65 005035  110203                          ROL     R3              ; ROTATE MASK TO CORRECT POSITION
     66 005036  110203                          ROL     R3              ; ROTATE MASK TO CORRECT POSITION
     67 005037  103203  177417                  BIC     #LBHINB,R3      ; CLEAR UNUSED BITS
     68 005041  100653  000045                  MOV     R3,U.WPRT(R5)   ; SAVE
     69 005043  104207  000001                  MOV     #SETUP,R0       ; SETUP NEXT ROUTINE CALLED
     70 005045  104051                          MOV     R5,R1           ; MAKE R1 NON-ZERO (DEFERRED CALL)
     71 005046  114002                          CLR     R2              ; NO ERRORS
     72 005047  003231                  DRVEXT: BR      JMPRET          ; RETURN TO         R
     73                                         .DSABL  LSB
     74                                         .IF     LE,MAXADR-.
     75                                 MAXADR  =       .+1
     76                                         .ENDC
     86 005050                                  DMEND
        005050  000105                          .WREDC          ;OUTPUT EDC FOR THIS OVERLAY
     87         000001                          .END
```

SYMBOL TABLE

```
AFTOP   004641   COPLP0  004557   D$      005524   ER44    003132   GETSUB= 000210
AFTWRT= 000011   COPLP1  004564   D.LIMT= 000001   ER45    003165   GETU    004764
ALLOCM  004703   COPLP2  004574   D.SCHR= 000002   ER47    003217   GMPARM  004710
AREAI = 004710   COUNTD  005064   ECC   = 000015   ER48    003246   GOINIT  004243
AREA0 = 004422   CPYBEX  006004   ECCCHK= 010000   ER49    003273   GONE    003202
AREA1 = 004611   CR.CLR  001624   ECCFLG= 010000   ER5     000365   GORCLB  004235
AREA2 = 005052   CR.DIS  001650   ECCRSH= 000002   ER50    003320   GORTRY  004221
ATTN  = 000002   CR.ERR  001643   ECHO  = 000010   ER51    003365   GOSEEK  004227
AVAIL = 000100   CR.GCR  004462   ECHOC = 000350   ER52    003410   GOTOBE  005103
BBLOP   005750   CR.GST  001617   EDCLOP  001155   ER53    003470   GOTOFS  005106
BEEXT   005260   CR.INR  001655   ENABLE  004203   ER54    003505   GO4IT   003214
BESDWN  005143   CR.MOD  001631   EOC   = 100000   ER55    003555   GROUP   002203
BESUP   005257   CR.ONL  001612   ERCOV = 000016   ER56    003607   GRPCYL= 000002
BEUSED= 000040   CR.RUN  004474   ERECOV= 000006   ER57    003647   GRPOFF= 000011
BF.DAT= 000000   CR.SCR  004467   EREXT   004513   ER58    003722   GRP$    005516
BF.ECC= 000401   CR.SEK  001636   ERHARD= 100000   ER59    003753   GST     001670
BF.EDC= 000400   CSCEXT  004517   ERLEV = 000002   ER6     000562   HBHINB= 007777
BLDBB   005555   CURBN   002175   ERMASK= 100000   ER62    003770   HBLONB= 170377
BLDUNT  005374   CVT   = 000020   ERMODE  002226   ER63    004120   HDRPRE= 000005
BLKCHK  001725   CYL     002201   ERR     001676   ER64    004150   HD.BAD= 110000
BREAK = 000000   CYLBN   005346   ERRLEV  001677   ER68    004200   HD.DBN= 140000
BSUSEX  006015   CYLSCR  005604   ERRMC = 060014   ER69    004223   HD.LBN= 000000
BUFARA= 005237   C2DFTL= 100000   ERRMES= 060013   ER7     000615   HD.PRV= 050000
BUFFLG= 040000   C2HARD= 040000   ERRPOS  002230   ER70    004264   HD.RBN= 060000
BUFSIZ= 000043   DATCMP= 000002   ERSOFT= 140000   ER71    004300   HD.REV= 030000
BUILDP= 000007   DATPRE= 000005   ER1     000000   ER72    004315   HD.XBN= 120000
BULDUM  001777   DBNCYL= 000022   ER10    000770   ER73    004345   HIBN    002172
CALC    002027   DCEXT   004752   ER1000  005071   ER74    004371   HIBYTE= 177400
CALCSC  002173   DCLOCK= 000004   ER11    001044   ER75    004416   HICYL = 000001
CBB2    004632   DCLR    001667   ER12    001116   ER76    004447   HIDBN = 000003
CHAINS  002232   DCMPAL= 000001   ER13    001411   ER77    004526   HILBN = 000002
CHGMOD= 000201   DCMPYS  004746   ER14    001440   ER78    004562   HIMEM = 007774
CHKBB   005061   DCREAD  004740   ER15    001506   ER8     000631   HIRBN = 000003
CHKBES  005420   DCYLS = 020000   ER16    001556   ER9     000745   HISEED  002225
CHKCYL  005243   DEBUG = 000000   ER17    001607   EXIT  = 000021   HIXBN = 000002
CHKDN   004104   DIE   = 000002   ER17A   001637   FB.DAT= 000000   HOSTRQ  001053
CHKECC= 000015   DINIT = 000011   ER18    001705   FB.EDC= 000400   HRDREV= 000003
CHKEDC= 000014   DIREC = 010000   ER19    001757   FCTSIZ= 000010   HRQ.RQ  001103
CHKTG   005232   DIS     001662   ER2     000117   FIRSTU= 007702   HRQ.01  001104
CHKUP   004134   DISCON= 000204   ER20    002061   FIXPAT  004702   HRQ.02  001105
CHNMAX  002234   DONE  = 060016   ER21    002111   FNDBES  005275   HRQ.03  001106
CHRRES= 000170   DOWNG   005351   ER23    002162   FNDLOP  005305   HRQ.04  001107
CLCLBN  005153   DOWNT   005207   ER24    002275   FNDRER  004650   HRQ.05  001110
CLCMAX  005746   DRC     001666   ER25    002342   FORMAT= 000001   HRQ.06  001111
CMPDAT= 000020   DRINIT= 040000   ER26    002371   FSTOP = 100000   HRQ.07  001112
CMPEDC  001146   DROP  = 100000   ER28    002420   FTIME = 001000   HRQ.08  001113
CMPERR  004524   DRPALL= 000026   ER3     000166   FTLDEV= 040000   HRQ.09  001114
CMP2    001765   DRTYPE= 000007   ER33    002455   FTLSYS= 000000   HRQ.10  001115
CMXEX   004716   DRVCLR= 000005   ER34    002543   FT.BUF= 000000   HRQ.11  001116
CNTLOP  005056   DRVEXT  005047   ER35    002566   FT.HI = 000001   HRQ.12  001117
COMCHR= 000030   DRVID = 000004   ER36    002615   FT.LOW= 000002   HRQ.13  001120
COMPAR= 000006   DRVONL= 000213   ER37    002637   FWRD    004523   HRQ.14  001121
COMPDP  005515   DRVRUN= 000014   ER38    002663   GCR     004502   HRQ.15  001122
COMPLT= 000176   DSABLE  004212   ER4     000271   GETCHR= 000207   HRQ.16  001123
COMPSC  005717   DSERNM  004525   ER40    002710   GETMEM  004625   HRQ.17  001124
COPBB   005730   DUMSDI  002207   ER41    003000   GETSER= 000035   HRQ.18  001125
COPFIN  004577   DUPPKT= 060000   ER42    003071   GETSTA= 000011   HRQ.19  001126
```

SYMBOL TABLE

```
HRQ.20  001127          MASK    005034          ONLYCL= 000200          OVL.SC= 000615          PAT13   002476
HRQ.21  001130          MAXADR= 006033          OTABLE  004266          OVL.SK= 000323          PAT14   002520
HRQ.22  001131          MAXDBN  004511          OUTO    005313          OVL.SO= 000222          PAT15   002525
HRQ.23  001132          MAXMSK= 177774          OVE.AW= 004422          OVL.SP= 000027          PAT2    002303
HRQ.24  001133          MAXMUM  005314          OVE.BP= 004422          OVL.ST= 000471          PAT3    002306
HRQ.25  001134          MAXSEC  002235          OVE.CC= 004710          OVL.SU= 000167          PAT4    002311
HRQ.26  001135          MAXSND= 001750          OVE.CD= 004422          OVL.S0= 001101          PAT5    002333
HRQ.27  001136          MAXSUB= 000010          OVE.CK= 004422          OVL.S1= 000624          PAT6    002355
HRQ.28  001137          MEDTYP= 000006          OVE.DA= 004422          OVL.TS= 000371          PAT7    002377
HRQ.29  001140          MEMPOL  002227          OVE.EC= 004422          OVL.W = 000404          PAT8    002402
HRQ.30  001141          MESSAG= 060015          OVE.ED= 004422          OVL...= 026744          PAT9    002424
HRQ.31  001142          MICREV= 000003          OVE.GS= 004710          OVSTRT= 007774          PHYSA = 034200
HRQ.32  001143          MINADR= 006232          OVE.ID= 004710          OVS.AW= 070624          PLOAD   003211
HRQ.33  001144          MKLOOP  004651          OVE.IN= 004710          OVS.BP= 067032          PNUM    002221
HRQ.34  001145          MOD     001664          OVE.LM= 004422          OVS.CC= 103650          PREVBE  005334
ID      005565          MOD512= 126736          OVE.MN= 000714          OVS.CD= 077210          PTABLE  004254
INDEX   002206          MOD576= 074161          OVE.MS= 000000          OVS.CK= 075420          RANDOM  004773
INITD = 000036          MORE    004722          OVE.NL= 004422          OVS.DA= 103456          RBLOCK  004546
INITW = 040000          MOVDBN  002123          OVE.NO= 004611          OVS.EC= 076456          RBNBN = 000200
INR     001671          MOVOUT  002164          OVE.R = 004422          OVS.ED= 074442          RBNFLG  002200
INS     001672          MOV2    005427          OVE.RB= 005052          OVS.GS= 110552          RBNTRK= 000004
INSEEK= 000012          MRD   = 000016          OVE.RC= 005237          OVS.ID= 111646          RBNTXT  005475
INSET = 000027          MSSG$ = 000000          OVE.RT= 005052          OVS.IN= 103604          RBUFLN= 000415
INTEDC= 000105          MS1     005532          OVE.RV= 004422          OVS.LM= 100012          RCBREQ= 002000
INTINP= 000001          MS2     005550          OVE.SB= 005052          OVS.MN= 034240          RCONT = 000000
IRECLB= 000216          MS3     005621          OVE.SC= 004422          OVS.MS= 046476          RCTCPS= 000001
ISUEXT  005560          MS4     005704          OVE.SK= 005237          OVS.NL= 076646          RCTCSZ= 000014
ISUSEK  005502          MWR   = 000017          OVE.SO= 004710          OVS.NO= 062760          RCTL$   005526
IWIPRG= 100000          M.PARM  002223          OVE.SP= 004710          OVS.R = 072234          RCV   = 000005
JMPRET  003231          NEWDNT  005433          OVE.ST= 005052          OVS.RB= 063462          RCVERR= 000400
LASTU   004506          NEWEXT  004600          OVE.SU= 004422          OVS.RC= 103316          RCVRDY= 000001
LBHINB= 177417          NEWLEV= 000017          OVE.S0= 004710          OVS.RT= 065320          RDSTAT  001007
LBLONB= 177760          NEWOP = 000002          OVE.S1= 004710          OVS.RV= 100146          READ  = 000012
LBNCYL= 000000          NEWSUB= 004000          OVE.TS= 005237          OVS.SB= 064022          RECALB= 000025
LBNHST= 000012          NEWUPT  005405          OVE.W = 004422          OVS.SC= 073010          RECOVR  003720
LBNTRK= 000011          NEXTBE  005370          OVL.AW= 000604          OVS.SK= 101466          REDWRT= 000100
LEVNZR  004540          NLBEXT  005144          OVL.BP= 000271          OVS.SO= 104234          RETRY = 001000
LEVUSD= 020000          NLEV    004502          OVL.CC= 000143          OVS.SP= 104156          RETS  = 000001
LINKLN= 000007          NOBB    006002          OVL.CD= 000301          OVS.ST= 065650          REVCT = 000022
LNCONT  002107          NOBORO  005120          OVL.CK= 000417          OVS.SU= 062402          REVEC = 000400
LNCYL   002101          NOEXTR  005765          OVL.DA= 000053          OVS.S0= 104700          REVS  = 000007
LOADED  003175          NOSEEK  005426          OVL.EC= 000074          OVS.S1= 107102          REVSOK  004534
LOBYTE= 000377          NOSUB   003715          OVL.ED= 000367          OVS.TS= 102334          RM    = 000004
LOCYL   001673          NOSUN   006013          OVL.GS= 000436          OVS.W = 067614          RNDBE = 000003
LONG  = 001654          NOU     005465          OVL.ID= 000141          OV... = 045064          RNDTG = 000005
LONGTO= 000001          NOUNIT  002560          OVL.IN= 000022          PART1   004264          RNDO    004676
LOSEED  002224          NUMBB   004510          OVL.LM= 000056          PART0   004254          RONLY = 004000
LRDTRK  002177          NUML2S= 000010          OVL.MN= 005117          PART1   004256          ROOT    002532
LSTMOD= 000021          NUMOVL= 000037          OVL.MS= 005742          PART2   004260          ROOTLP  002535
LSTOVL  002222          NUMPTR= 000006          OVL.NL= 000161          PART3   004262          RREAL = 013400
LSTTRK  005520          NXTLEV= 010000          OVL.NO= 000241          PATEXT  004704          RSTOP = 100000
L$      005522          NXTRCT  005003          OVL.R = 000266          PATPTR  002236          RTDS    000746
L2.EOF= 000004          NXTTRK  005442          OVL.RB= 000160          PAT0    002256          RTDSL   000775
L2.ERS= 000003          OCNT$ = 000037          OVL.RC= 000060          PAT1    002300          RTRIES= 001000
L2.OPC= 000000          ONEPAT  004573          OVL.RT= 000154          PAT10   002427          RUN     004501
L2.RLN= 000002          ONEPAX  004510          OVL.RV= 000550          PAT11   002451          RVFAIL  005134
L2.SLN= 000001          ONL     001700          OVL.SB= 000537          PAT12   002454          RWRDY = 100000
```

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE 143-4
SYMBOL TABLE

```
RW.ANG= 000006          SER17   005361          START   004530          TG$     005514          U.MLEV= 000031
RW.BUF= 000001          SER18   005413          STATUS= 000007          THREBE  005100          U.MSEC= 000022
RW.CMD= 000004          SER18A  005435          STOP    002573          TLEN.U= 000072          U.MSTO= 000011
RW.HI = 000003          SER18B  005431          STSERR  005426          TLKHST  001031          U.NEXT= 000000
RW.LOW= 000002          SER18C  005425          STSEXT  005626          TMEMPL  004710          U.NFUN= 000013
RW.SDI= 000005          SER18D  005421          STSRES= 000366          TO      005040          U.NSEC= 000021
RW.STA= 000000          SER18E  005350          ST.C  = 000002          TRACK   002204          U.PARM= 000046
SBCRES= 000167          SER18F  004470          ST.CON= 000002          TRACKS= 000020          U.PAT = 000014
SCHAR0= 000033          SER19   000470          ST.DB = 001000          TRAV    005165          U.PCTG= 000017
SCHAR1= 000034          SER2    004670          ST.DE = 000200          TRKGRP= 000003          U.RBN = 000055
SCHRCP  005217          SER20   004052          ST.DF = 000020          TRK$    005512          U.RCOV= 000047
SCR     004503          SER21   000661          ST.DR = 000040          TROOT   004647          U.RRTY= 000010
SCR1    002217          SER22   000051          ST.EL = 000010          TRYAGN  005066          U.RTRY= 000030
SCR2    002220          SER24   001331          ST.ERR= 000002          TSTNEC  005433          U.RVER= 000062
SCTWRD= 000377          SER25   001357          ST.FO = 002000          T1MSIZ= 060000          U.RWER= 000061
SDIVER= 000000          SER26   003040          ST.MOD= 000001          T2CMD = 060002          U.RWTO= 000012
SDI$  = 000003          SER3    004712          ST.MSK= 000000          T2DLL = 060001          U.SDIL= 000034
SEARCH  004644          SER32   003054          ST.OA = 000200          T4BB1 = 060005          U.SDIS= 000033
SECCHK= 000013          SER36   004245          ST.PE = 000040          T4BB2 = 060006          U.SDI2= 000035
SECCYL  004515          SER39   002502          ST.PS = 000002          T4MPRM= 060003          U.SRTY= 000007
SECGRP  004514          SER4    004731          ST.RE = 000100          T4MXFR= 060011          U.SUBP= 000001
SECMAX  002233          SER40   005440          ST.REQ= 000001          T4SEEK= 060010          U.SUBU= 000050
SECPTR  004711          SER41   005462          ST.RR = 000100          T4SOFT= 060007          U.TIMH= 000005
SECTOR  002205          SER42   002203          ST.RTY= 000003          T4UPRM= 060004          U.TIML= 000006
SECTRK  004513          SER43   002217          ST.RU = 000001          UCURSR  004505          U.TSEC= 000023
SEEK  = 000023          SER44   002234          ST.SR = 000020          UDADM4= 034200 G        U.UNUM= 000063
SEKCNT  005304          SER45   002246          ST.S7 = 000400          ULOP    005402          U.WPRT= 000045
SEKEXT  005432          SER5    004752          ST.UNT= 000000          UMASK   004507          U.WRIT= 000026
SEKINP= 002000          SER6    004773          ST.WE = 000010          UNITS   005354          VALBIT= 077177
SEKOUT  005431          SER7    005013          SUBTRV  005176          UNSSUC= 000175          VALID = 100000
SEKREQ= 004000          SER8    005032          SUBUNT  004504          UP      005163          WAITSI= 000012
SEKTST= 000024          SER9    005053          SUNIT   005335          UPG     005271          WBLOCK  004646
SEND  = 000004          SETSEC  005507          SUSETL  005505          UPT     005130          WBUFLN= 000401
        000004          SETUP = 000001          SWAPBE  005004          UREAD = 000013          WCHECK= 000010
        002601          SHRTTO= 000000          S.BADP= 000012          UTOTST= 060012          WCHKAL= 000004
        002577          SLOP    005516          S.BESS= 000013          UWRITE= 000014          WCONT = 040000
        000100          SMASKL  004721          S.LETR= 000005          U.CBN = 000053          WONLY = 002000
        000006          SMASKX  004725          S.MCNT= 000013          U.CCNT= 000015          WREAL = 122400
SERCHN  005337          SNDAGN  001060          S.MEGR= 000010          U.CCOP= 000060          WRITBT  001665
SERRTY  005336          SNDEX   004601          S.MEGW= 000011          U.CCYL= 000064          WRITE = 000010
SERSEC  005345          SNDONE= 001616          S.PARM= 000000          U.CGRP= 000066          WSTOP = 140000
SER0    004633          SORT  = 000032          S.PAT = 000004          U.COPY= 000057          XBNCYL= 000021
SER1    004651          SORTBB  005034          S.SCHR= 000007          U.CSEC= 000024          XFERRT= 000000
SER10   005210          SORTBE  004744          S.SDCL= 000002          U.CTRK= 000020          XOPLP0  004471
SER11   005223          SORTTG  005103          S.SEEK= 000001          U.ECCT= 000032          XOPLP1  004476
SER12   005241          SPINUP= 000031          S.TGOF= 000015          U.ELEV= 000027          XOPLP2  004511
SER13   005253          SS    = 000001          S.TGSS= 000017          U.LCYL= 000067          XREAD = 000002
SER14   005266          ST      001702          S.TRKL= 000006          U.LGRP= 000071          XWRITE= 000003
SER15   005314          STACK   000745          TALK    001173          U.MASK= 000025          ZEREDC  001165
SER16   005336          STACYL  002173          TGS     004516          U.MBN = 000051

. ABS.  112150      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  6098 WORDS  ( 24 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  71 PAGES
,B:UDAT4/C=$DMACRO,B:UDAT4
```

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE S-1
CROSS REFERENCE TABLE (CREF V04.00)

```
AFTOP   63-2#    72-68    73-128   83-17    84-48
AFTWRT  44-39    97-112   99-10#   100-10
ALLOCM  94-14    94-17    94-28    95-21    96-2#
AREA0   44-9     44-57#   61-2     61-2     93-2     93-2     97-2     97-2     99-2     99-2     100-2    100-2    103-2    103-2
        104-2    104-2    105-2    105-2    106-2    106-2    107-2    107-2    108-2    108-2    109-2    109-2    110-2    110-2
        120-2    120-2
AREA1   44-11    61-107#  62-2     62-2
AREA2   44-13    70-17#   71-2     71-2     73-2     73-2     80-2     80-2     84-2     84-2
AREAI   44-17    49-27#   121-2    121-2    122-2    122-2    123-2    123-2    124-2    124-2    129-2    129-2    136-2    136-2
        141-2    141-2    143-2    143-2
ATTN     8-23#   18-18    34-46    36-18    36-25    36-27    38-87    39-11    52-14    118-72
AVAIL    8-26#   18-18    34-46    36-16    36-23    38-51    51-34    118-65   119-39
BBLOP   58-86#   58-100
BEEXT   73-69    73-121   73-123#
BESDWN  73-36    73-59#
BESUP   73-102   73-122#
BEUSED  10-52#   56-9     58-14    58-25    62-19    62-25    124-25   129-80   133-17   133-43   133-61   136-42
BF.DAT   5-62#
BF.ECC   5-64#
BF.EDC   5-63#   97-97*   104-56   104-83   105-47   105-73   141-127*
BLDBB   56-13    57-2#
BLDUNT  46-3     55-2#
BLKCHK  25-2#    99-61    103-30   104-31   108-76
BREAK    6-3#
BSUSEX  58-110   58-112#
BUFARA  44-15    44-107   44-115   46-15    96-12    98-46    99-155   102-26   103-142  104-106  105-101  106-52   107-69   108-111
        109-46   113-49   115-2    115-2    118-2    118-2    119-2    119-2
BUFFLG   5-47#   99-37    102-13   103-28
BUFSIZ  20-15    20-20    21-41#
BUILDP  44-37    93-10#   97-10    106-41   107-64   115-69   118-58
BULDUM  27-2#    97-43    100-51   141-32
C2DFTL   8-65#   23-96    122-21   123-22   129-134
C2HARD   8-64#   104-94   105-82
CALC    28-2#    79-17    79-29    93-28    113-25   116-16
CALCSC  28-60    29-3#    141-50
CBB2    48-2#    127-12   133-13   133-28   133-39   133-57   137-14   137-26   137-40
CHAINS  29-37#   93-27*   94-7     94-18*   97-62    98-7     99-32    101-6    102-9    103-25   104-24   104-75   105-22   108-23
        108-79   109-20   109-26*  112-6
CHGMOD   8-34#   24-22
CHKBB   136-37   137-2#
CHKBES  129-87   133-2#
CHKCYL  129-84   131-2#
CHKDN   33-121   33-125   41-2#
CHKECC  44-43    104-42   105-10#  106-10
CHKEDC  44-42    103-137  104-10#  105-10
CHKTG   136-44   138-2#
CHKUP   33-79    33-118   33-128   42-2#
CHNMAX  29-39#   46-29*   61-86    61-88
CHRRES   8-45#   44-74
CLCLBN  72-27    72-33#
CLCMAX  129-77   135-2#
CMP2    25-15    26-2#    73-35    74-13
CMPDAT  44-46    104-73   104-98   105-95   107-43   108-10#  109-10
CMPEDC  22-2#    97-96    104-54   105-46   141-126
CMPERR  108-56   108-65   108-73#
CMXEX   108-71   108-78   108-108#
```

CROSS REFERENCE TABLE (CREF V04.00)

```
CNTLOP  71-20#   71-24
COMCHR  44-59   121-23   122-10#  123-10
COMPAR   6-9#
COMPDP 139-10   139-31   139-49   140-2#
COMPLT   8-43#   24-6     24-9     24-10    24-11    24-12    24-13    24-15    44-76
COMPSC 129-76   134-2#
COPBB   58-52    58-76#
COPFIN  97-82    97-90#
COPLP0  97-75#   97-85
COPLP1  97-79#   97-84
COPLP2  97-87#   97-89
COUNTD  71-21    71-25#
CPYBEX  58-104   58-106#
CR.CLR  24-9#    40-52
CR.DIS  24-13#   31-40
CR.ERR  24-12#  107-49
CR.GCR  44-74#  122-16
CR.GST  24-8#    38-54    40-15    51-40
CR.INR  24-15#  119-22   119-41   141-56
CR.MOD  24-10#   40-62
CR.ONL  24-6#    40-9
CR.RUN  44-76#  123-15
CR.SCR  44-75#  129-35
CR.SEK  24-11#  117-9    141-63
CSCEXT 108-59   108-69#
CURBN   29-5#    79-13*   79-15*   79-57    79-59    79-72*   79-74*   79-76*   79-78*   93-20*   93-21*   93-23*   93-24*   95-31
        95-33    95-54*   95-56*  110-56*  110-60*  111-17   111-17   112-14   112-16*  112-51   112-51   113-22*  113-23*  113-41
       113-41   116-10*  116-11*  116-13*  116-14*  141-48*  141-49*  141-90   141-99   141-135* 141-137*
CVT     6-19#    28-62   141-52
CYL    29-8#    79-19    79-31    93-29*   93-31*  113-27*  113-30*  115-43   115-43   116-17   117-6    117-7    117-24   141-60
       141-61
CYLBN  131-11   131-24   132-2#
CYLSCR  79-18    79-30    79-82#
D$     59-286#  129-65   133-65   133-66
D.LIMT  10-4#    27-15*
D.SCHR  10-5#    27-18*
DATCMP  10-57#   58-14    63-14    67-6     67-17   104-69   104-96   105-93   107-41
DATPRE   7-32#   98-10
DBNCYL   7-41#  135-10   135-18
DCEXT   67-7     67-11    67-16    67-19#
DCLOCK   8-24#   18-33
DCLR    24-25#   40-51*
DCMPAL  10-59#   58-14    67-12
DCMPYS  67-13    67-17#
DCREAD  67-9     67-12#
DCYLS   10-44#   27-8     28-17    58-14    58-20    95-38    99-94   103-60   129-63   131-35   133-36   137-22   141-153
DEBUG    2-37#    3-2     31-3     34-52    35-6     44-95    47-9     59-291   61-18    61-30    97-15    99-16   100-15   103-18
       104-18   105-15   107-15   108-15   110-12   115-19   118-17   119-15
DIE     10-10#   46-6    122-23   123-24   129-104  129-136  136-51   141-159  143-56
DINIT    6-12#   43-53
DIREC   10-17#   33-14    33-116   41-23    42-35    73-16    79-54    79-70    84-18
DIS     24-13    24-20#
DISCON   8-32#   24-20
DONE     8-18#   31-52
DOWNG   84-24    88-2#
DOWNT   84-22    86-2#    88-23
```

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

```
        99-86    99-117   99-122   99-126   99-131   99-136   103-52   103-86   103-90   103-95   103-100  103-104  103-109  103-114
       104-44    104-78   105-29   105-35   105-68   118-54   118-75   118-91   119-32
EXIT     6-20#    31-54
FB.DAT   5-22#
FB.EDC   5-23#
FCTSIZ   7-35#   141-135
FIRSTU   9-79#    31-24    45-16    49-7     49-24    55-49    55-52    58-120   129-44   129-46
FIXPAT  65-11     65-15#
FNDBES  73-33     73-101   74-2#
FNDLOP  74-11#    74-15
FNDRER 101-17    102-2#
FORMAT   6-4#
FSTOP    5-42#
FT.BUF   5-15#
FT.HI    5-16#
FT.LOW   5-17#
FTIME   10-20#    33-11    33-14    39-47    42-33    49-10    49-20
FTLDEV   8-61#     8-65     8-65     18-9     34-23    36-20    38-3     38-4     40-25    46-8     54-38    98-15    98-36    101-11
       115-42    132-22   133-21   133-23   133-30   133-53   133-67   133-69   136-23   137-18   137-36   137-42   138-45   139-29
       141-155   143-54
FTLSYS   8-60#
FWRD    108-43   108-72#
GCR      44-74     44-85#
GETCHR   8-38#    44-85
GETMEM  47-2#     55-23    57-15    129-54
GETSER  44-64    129-131  136-78   141-10#  143-10
GETSTA   8-40#    24-26
GETSUB   8-39#    44-86
GETU    46-2      51-2#
GMPARM  46-1      50-2#
GO4IT   33-142    34-40#
GOINIT  23-23     23-55    43-48#   99-87    99-127   99-132   103-53   103-105  103-110
GONE    34-33#    34-37
GORCLB  40-40     43-39#   99-118   103-96   115-48
GORTRY  23-35     23-88    34-48    43-21#   119-37
GOSEEK  33-13     40-50    43-30#   43-54    61-101   98-17    98-38    99-148   101-13   103-126  118-94
GOTOBE  71-30     71-33    71-36#
GOTOFS  71-26     71-39#
GROUP   29-9#     93-33*   115-43   117-8    141-62
GRP$    59-282#  138-33
GRPCYL   7-24#   134-15   138-32   139-42
GRPOFF   7-37#
GST     24-8      24-26#
HBHINB   7-49#    26-11    28-38    28-50    48-7     48-9     52-10    55-31    58-119   73-51    73-79    76-26    77-27    78-11
        94-23     99-43   102-18   108-38   109-25   110-47   112-13   125-16   132-20   138-22
HBLONB   7-50#    28-29    28-33    28-58    97-60    141-97
HD.BAD   5-55#
HD.DBN   5-58#    95-40
HD.LBN   5-52#    95-44
HD.PRV   5-56#
HD.RBN   5-53#    95-42
HD.REV   5-54#
HD.XBN   5-57#   141-98
HDRPRE   7-33#
HIBN    28-34*    28-59*   29-2#    95-43
HIBYTE   7-47#    27-12    28-11    28-13    28-20    28-44    38-91    40-46    82-12    84-42    91-8     98-11    122-36   122-40
```

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE S-6
CROSS REFERENCE TABLE (CREF V04.00)

```
        126-18   127-27   128-14   129-67   129-73   134-9    134-16   135-12   135-19   138-37   138-41   139-9    139-25   139-46
        139-47   141-37   141-39
HICYL     7-23#    28-37    28-49   113-27
HIDBN     7-29#    28-53
HILBN     7-25#    28-32
HIMEM     5-9#      9-79    29-26    44-105
HIRBN     7-28#    28-28
HISEED   29-24#    69-9     69-19    69-25*   72-23    80-23
HIXBN     7-26#   141-92
HOSTRQ   19-7      20-2#    23-106   31-53    33-77    34-26    37-6     39-4     39-13    46-11    50-25    53-5     54-40    56-8
         57-9      58-9     58-79    58-91    98-43    99-116   101-22   103-94   104-63   105-62   105-65   115-51   118-46   118-53
        118-74   120-40   129-103  136-30   136-50   141-158  141-169
HRDREV    7-13#
HRQ.01   18-10*   19-7*    21-6#    23-24*   23-40*   23-56*   23-60*   23-64*   23-68*   23-72*   23-75*   23-82*   23-106*  33-77*
         33-77*   34-23*   36-20*   36-28*   37-3*    37-4*    37-7*    39-5*    39-13*   40-25*   46-8*    50-26    53-6     54-38*
         56-6*    56-9*    57-7*    57-11    58-7*    58-16*   58-19    58-77*   58-89*   58-93    97-33*   98-15*   98-36*   98-43*
         99-86*   99-101*  99-116*  99-117*  99-122*  99-126*  99-131*  99-136*  99-138*  100-36*  101-11*  101-22*  103-52*  103-68*
        103-86*  103-90*  103-94*  103-95*  103-100*  103-104*  103-109*  103-114*  103-116*  104-44*  104-63*  104-78*  105-29*  105-35*
        105-60*  105-65*  105-65*  105-68*  107-31*  108-80*  111-10*  112-44*  113-34*  115-42*  115-51*  118-43*  118-44*  118-53*
        118-54*  118-74*  118-75*  118-91*  120-32*  132-22*  133-21*  133-23*  133-30*  133-53*  133-67*  133-69*  136-23*  136-30*
        136-30*  137-18*  137-36*  137-42*  138-45*  139-29*  141-155*  141-160*  143-54*
HRQ.02   18-9*    19-7*    21-7#    23-24*   23-40*   23-56*   23-60*   23-64*   23-68*   23-72*   23-75*   23-82*   23-96*   23-106*
         33-77*   34-23*   36-20*   36-28*   38-1     39-13*   40-25*   46-8*    54-38*   58-17    58-81    97-33*   97-106*  98-15*
         98-36*   98-43*   99-86*   99-101*  99-116*  99-117*  99-122*  99-126*  99-131*  99-136*  99-138*  100-36*  101-11*  101-22*
        103-52*  103-68*  103-86*  103-90*  103-94*  103-95*  103-100*  103-104*  103-109*  103-114*  103-116*  104-44*  104-63*  104-78*
        104-94*  105-29*  105-35*  105-53*  105-56*  105-65*  105-68*  105-82*  107-31*  108-80*  109-37*  111-10*  112-44*  113-34*
        115-42*  115-51*  118-53*  118-54*  118-74*  118-75*  118-91*  120-20*  122-21*  123-22*  129-134*  132-22*  133-21*  133-23*
        133-30*  133-53*  133-67*  133-69*  136-23*  136-30*  137-18*  137-36*  137-42*  138-45*  139-29*  141-155*  141-161*  143-54*
HRQ.03   19-7*    21-8#    23-106*  34-24*   39-1*    39-2*    39-13*   46-9*    54-5*    56-15    58-31    58-55    97-107*  98-43*
         99-116*  101-22*  103-94*  104-63*  105-49*  105-65*  109-38*  115-51*  118-53*  118-74*  120-38*  129-100*  129-101*  136-47*
        136-48*  141-156*  141-162*
HRQ.04   18-6*    18-8*    19-7*    21-9#    23-24*   23-40*   23-56*   23-60*   23-64*   23-68*   23-72*   23-75*   23-82*   23-106*
         34-23*   36-20*   36-28*   39-13*   40-25*   46-8*    54-38*   97-33*   98-15*   98-36*   98-43*   99-86*   99-101*  99-116*
         99-117*  99-122*  99-126*  99-131*  99-136*  99-138*  100-36*  101-11*  101-22*  103-52*  103-68*  103-86*  103-90*  103-94*
        103-95*  103-100*  103-104*  103-109*  103-114*  103-116*  104-44*  104-63*  104-78*  105-29*  105-35*  105-65*  105-68*  107-31*
        108-80*  111-10*  112-44*  113-34*  115-42*  115-51*  118-53*  118-54*  118-74*  118-75*  118-91*  120-21    132-22*  133-21*
        133-23*  133-30*  133-53*  133-67*  133-69*  136-23*  137-18*  137-36*  137-42*  138-45*  139-29*  141-155*  141-163*  143-54*
HRQ.05   21-10#   34-23*   36-29*   40-13*   40-19*   40-28*   40-32*   40-34*   40-56*   40-64*   46-8*    54-6     97-33*   98-15*
         98-36*   99-102*  99-138*  99-141*  100-36*  101-11*  103-69*  103-116*  103-119*  104-44*  104-78*  105-36*  105-65*  105-68*
        107-31*  107-53*  108-81*  108-84*  111-14*  111-16*  112-48*  112-50*  113-38*  113-40*  115-42*  117-33*  118-54*  118-75*
        118-91*  119-48*  122-20*  123-21*  129-133*  132-22*  133-21*  133-23*  133-30*  133-53*  133-67*  133-69*  137-18*  137-36*
        137-42*  138-45*  139-29*  141-155*  141-164*  143-54*
HRQ.06   21-11#   23-73*   23-76*   23-83*   34-23*   40-35*   97-33*   98-15*   98-37*   99-102*  99-138*  99-142*  100-36*  101-11*
        103-69*  103-116*  103-120*  104-44*  104-78*  105-36*  105-65*  105-68*  107-31*  108-85*  111-17*  112-51*  113-41*  115-42*
        118-54*  118-75*  118-91*  133-47*  133-53*  133-65*  133-69*  137-36*  137-42*  138-45*  139-29*  141-165*
HRQ.07   21-12#   23-77*   23-83*   56-12    58-51    97-33*   98-15*   99-102*  99-139*  99-142*  100-36*  101-11*  103-69*  103-117*
        103-120*  104-44*  104-78*  105-36*  105-65*  105-68*  107-31*  108-85*  111-17*  112-51*  113-41*  115-42*  118-54*  118-75*
        118-91*  133-48*  133-53*  133-66*  133-69*  137-36*  137-42*  138-45*  141-166*
HRQ.08   21-13#   23-83*   97-34*   98-16*   99-103*  99-142*  100-37*  101-12*  103-70*  103-120*  104-45*  104-79*  105-37*  105-65*
        105-69*  107-32*  108-85*  111-17*  113-41*  115-43*  118-54*  118-75*  118-92*  141-167*
HRQ.09   21-14#   97-34*   99-103*  99-142*  100-37*  103-70*  103-120*  104-45*  104-79*  105-37*  105-65*  105-69*  107-32*  108-86*
        111-17*  113-41*  115-43*  118-55*  118-75*  118-92*
HRQ.10   21-15#   97-34*   99-103*  99-143*  100-37*  103-70*  103-121*  104-45*  104-79*  105-37*  105-65*  105-69*  107-32*  108-86*
        115-43*  118-55*  118-76*  118-92*
HRQ.11   21-16#   97-34*   99-104*  99-143*  100-37*  103-71*  103-121*  104-46*  104-80*  105-38*  105-70*  107-32*  108-87*  115-43*
        118-55*  118-77*  118-93*
```

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE S-7
CROSS REFERENCE TABLE (CREF V04.00)

```
HRQ.12  21-17#   99-104*  99-143* 103-71*  103-121* 104-46*  104-80*  105-38*  105-70*  107-32*  108-87*  115-43*  118-55*
HRQ.13  21-18#   99-105*  99-144* 103-72*  103-122* 104-47*  104-81*  105-39*  105-71*  107-33*  108-87*  115-43*  118-55*
HRQ.14  21-19#   99-105*  99-144* 103-72*  103-122* 104-47*  104-81*  105-39*  105-71*  107-33*  108-88*  115-43*  118-55*
HRQ.15  21-20#   99-105*  99-145* 103-72*  103-123* 104-47*  104-81*  105-39*  105-71*  108-88*  118-56*
HRQ.16  21-21#   99-105*  99-145* 103-72*  103-123* 104-48*  104-82*  105-40*  105-72*  108-88*
HRQ.17  21-22#   99-106*  99-145* 103-73*  103-123* 104-48*  104-82*  105-40*  105-72*  108-88*
HRQ.18  21-23#   99-106*  99-145* 103-73*  103-123* 104-48*  104-82*  105-40*  105-72*  108-105*
HRQ.19  21-24#   99-106*  99-146* 103-73*  103-124* 104-49*  104-83*  105-41*  105-73*  108-105*
HRQ.20  21-25#   99-146* 103-124* 104-83*  105-73*  108-105*
HRQ.21  21-26#   99-146* 103-124* 104-83*  105-73*  108-105*
HRQ.22  21-27#   99-147* 103-125* 108-105*
HRQ.23  21-28#  108-105*
HRQ.24  21-29#  108-105*
HRQ.25  21-30#  108-105*
HRQ.26  21-31#  108-105*
HRQ.27  21-32#  108-105*
HRQ.28  21-33#  108-105*
HRQ.29  21-34#  108-106*
HRQ.30  21-35#  108-106*
HRQ.31  21-36#  108-106*
HRQ.32  21-37#  108-106*
HRQ.33  21-38#
HRQ.34  21-39#
HRQ.RQ  18-11*   20-13*   20-14    20-19    20-22*   21-5#    21-41    23-24*   23-40*   23-56*   23-60*   23-64*   23-68*   23-72*
        23-75*   23-82*   34-23*   34-25    36-20*   36-28*   37-1*    37-5     38-11    39-3     40-25*   46-8*    46-10    54-38*
        54-39    97-33*   97-109*  98-15*   98-36*   99-86*   99-101*  99-117*  99-122*  99-126*  99-131*  99-136*  99-138*  100-36*
       101-11*  103-52*  103-68*  103-86*  103-90*  103-95*  103-100* 103-104* 103-109* 103-114* 103-116* 104-44*  104-78*  104-95*
       105-29*  105-35*  105-68*  105-83*  107-31*  108-80*  109-40*  111-10*  112-44*  113-34*  115-42*  118-54*  118-75*  118-91*
       122-22*  123-23*  129-102  129-135* 132-22*  133-21*  133-23*  133-30*  133-53*  133-67*  133-69*  136-23*  136-49   137-18*
       137-36*  137-42*  138-45*  139-29*  141-155* 141-157  143-54*
ID      79-27    79-47    79-69#
INDEX   29-12#   95-46
INITD   44-65   141-177  143-10#
INITW   10-16#   31-34    33-32    33-74    33-85    42-22    42-32    58-14    58-24    61-55    62-17    64-8     66-10    87-11
       136-25   136-27   143-28   143-34
INR     24-15    24-27#
INS     24-11    24-28#  117-6*   117-7*   117-8*
INSEEK   8-42#   24-28
INSET   44-58    49-12   121-10#  122-10
INTEDC   9-77#   22-9
INTINP  10-11#   39-38    46-4     46-13
IRECLB   8-41#   24-27
ISUEXT 117-32   117-34#
ISUSEK 115-55   117-2#
IWIPRG  10-9#    31-33    31-49    31-50    33-31    33-73    42-21    42-31    75-9     79-11    84-36   143-36
JMPRET  34-49    35-4#    61-105   62-32    68-10    72-71    73-131   83-20    84-51    95-68    97-114   99-153  100-55   103-140
       104-104  105-99   106-50   107-68   108-110  109-44   110-71   115-72   118-95   119-60   121-24   122-67  123-26   124-38
       129-140  136-83   141-179  143-72
L$      59-284# 114-18   129-69   133-47   133-48
L2.EOF   8-56#   23-90    23-101   119-41
L2.ERS   8-55#    8-56    23-80    23-83
L2.OPC   8-52#    8-53    23-14    38-57    51-42
L2.RLN   8-54#    8-55    23-43    38-67    51-58
L2.SLN   8-53#    8-54    23-15    38-58    51-43
LASTU   45-16#   55-24*   55-26*   55-50*
LBHINB   7-51#   40-43   143-67
```

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE S-8
CROSS REFERENCE TABLE (CREF V04.00)

```
LBLONB    7-52#   52-24   65-10   65-13   79-49   97-60  108-26  108-45  122-33  122-43  122-47  122-61
LBNCYL    7-22#  113-30  129-90  129-93  141-34  141-35
LBNHST    7-38#  110-42  110-44  131-37  131-40  133-38  133-50* 133-52* 133-53  133-53  137-25
LBNTRK    7-36#   28-12   28-19   28-43  129-72  141-38
LEVNZR  107-27   107-29  107-48#
LEVUSD   10-33#   61-99   61-102  107-59  118-60
LINKLN    5-70#   46-18   46-24   46-26   94-16   94-27
LNCONT   28-30    28-34#
LNCYL    28-23    28-31#
LOADED   33-149   34-30#
LOBYTE    7-48#
LOCYL    24-29#  141-60* 141-61* 141-62*
LONG     23-28    24-14#
LONGTO    7-6#   122-42
LOSEED   29-23#   69-8    69-16   69-24*  72-18   80-19
LRDTRK   28-27*   28-31*  28-46*  29-6#   141-40*
LSTMOD   44-47   103-135 104-36   104-71  104-100 105-97  107-45  108-108 109-10# 110-10
LSTOVL   29-21#  33-140*  39-28
LSTTRK   88-9     88-24   92-2#
M.PARM   29-22#   31-31*  31-47*  31-50*  33-29*  33-71*  39-38*  42-19*  42-29*  46-4*   46-6*   46-13*  75-7*   79-9*
         84-34*  122-23* 123-24* 129-104* 129-136* 136-51* 141-159* 143-36* 143-56*
MASK     70-2#    72-22   72-29   80-22   80-29   82-21   83-10
MAXADR   44-93    58-123  70-18   72-73   79-86   83-22   92-19   96-15   98-49   99-158  102-29  103-145 104-110 105-104
        106-55   107-72  108-114 109-49  113-52  117-35  118-96  119-62  120-53  121-26  122-82  123-28  128-26  135-32
        140-19   142-11  143-74
MAXDBN   45-19#  131-45  131-47  133-56  133-69  133-69  135-28* 135-29* 137-39  137-42  137-42
MAXMSK    9-80#   72-57   83-7
MAXMUM   73-56    73-120  75-2#
MAXSEC   29-40#   93-17*  95-52*
MAXSND    9-81#   23-12   38-53   51-39
MAXSUB   44-101#  44-105  58-132  121-31  122-87  123-33  128-31
MEDTYP    7-34#
MEMPOL   29-26#   46-14   47-6*   47-7    93-25
MESSAG    8-17#   33-77   105-65  120-39  136-30
MICREV    7-12#
MINADR   44-105#  44-107  58-127  58-132  61-109  61-110  70-22   70-23   72-77   72-78   79-90   79-91   83-26   83-27
         92-23    92-24   96-19   96-20   98-53   98-54   99-162  99-163  102-33  102-34  103-149 103-150 104-114 104-115
        105-108  105-109 106-59  106-60  107-76  107-77  108-118 108-119 109-53  109-54  113-56  113-57  117-39  117-40
        118-100  118-101 119-66  119-67  120-57  120-58  121-30  121-31  122-86  122-87  123-32  123-33  128-30  128-31
        135-36   135-37  140-23  140-24  142-15  142-16  143-78  143-79
MKLOOP   49-8#    49-25
MOD      24-10    24-22#
MOD512    6-26#  141-149
MOD576    6-27#
MORE     50-11#   50-23
MOV2     76-18    77-18   78-2#
MOVDBN   28-18    28-41#
MOVOUT   28-7     28-40   28-60#
MRD       6-17#   20-16
MS1      33-77    60-5#
MS2      60-7#   120-20
MS3      60-10#  105-65
MS4      60-13#  136-30
MSSG$    44-28#   44-30
MWR       6-18#   20-21
NEWDNT   86-8     88-8    90-2#
```

CROSS REFERENCE TABLE (CREF V04.00)

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE S-10
CROSS REFERENCE TABLE (CREF V04.00)

```
        105-41   105-41   105-41#  105-65   105-65   105-65   105-65   105-65   105-65   105-65   105-65   105-65   105-65   105-65
        105-65   105-65   105-65   105-65   105-65   105-65#  105-65#  105-65#  105-65#  105-65#  105-65#  105-65#  105-65#  105-65#
        105-68   105-68   105-68   105-68   105-68   105-68   105-68#  105-68#  105-68#  105-68#  105-69   105-69   105-69   105-69
        105-69   105-69   105-69#  105-69#  105-69#  105-70   105-70   105-70   105-70   105-70#  105-70#  105-71   105-71   105-71
        105-71   105-71   105-71   105-71#  105-71#  105-71#  105-72   105-72   105-72   105-72   105-72   105-72   105-72#  105-72#
        105-72#  105-73   105-73   105-73   105-73   105-73   105-73   105-73#  105-73#  105-73#  107-31   107-31   107-31   107-31
        107-31   107-31   107-31#  107-31#  107-31#  107-31#  107-32   107-32   107-32   107-32   107-32   107-32   107-32   107-32
        107-32   107-32   107-32#  107-32#  107-32#  107-32#  107-32#  107-33   107-33   107-33   107-33   107-33#  107-33#  107-53
        107-53   107-53#  107-53#  108-80#  108-81   108-81   108-81#  108-81#  108-84   108-84   108-84#  108-84#  108-85   108-85
        108-85   108-85   108-85   108-85   108-85#  108-85#  108-85#  108-86   108-86   108-86   108-86   108-86#  108-86#  108-87
        108-87   108-87   108-87   108-87   108-87   108-87#  108-87#  108-87#  108-88   108-88   108-88   108-88   108-88   108-88
        108-88   108-88   108-88#  108-88#  108-88#  108-88#  108-105  108-105  108-105  108-105  108-105  108-105  108-105  108-105
        108-105  108-105  108-105  108-105  108-105  108-105  108-105  108-105  108-105  108-105  108-105  108-105  108-105  108-105
        108-105# 108-105# 108-105# 108-105# 108-105# 108-105# 108-105# 108-105# 108-105# 108-105# 108-105# 108-106  108-106  108-106
        108-106  108-106  108-106  108-106  108-106  108-106# 108-106# 108-106# 108-106# 111-10#  111-14   111-14   111-14#  111-14#
        111-16   111-16   111-16#  111-16#  111-17   111-17   111-17   111-17   111-17   111-17   111-17   111-17   111-17#  111-17#
        111-17#  111-17#  112-44#  112-48   112-48   112-48#  112-48#  112-50   112-50   112-50#  112-50#  112-51   112-51   112-51
        112-51   112-51#  112-51#  113-34#  113-38   113-38   113-38#  113-38#  113-40   113-40   113-40#  113-40#  113-41   113-41
        113-41   113-41   113-41   113-41   113-41   113-41   113-41#  113-41#  113-41#  113-41#  115-42   115-42   115-42   115-42
        115-42   115-42   115-42#  115-42#  115-42#  115-42#  115-43   115-43   115-43   115-43   115-43   115-43   115-43   115-43
        115-43   115-43   115-43   115-43   115-43   115-43   115-43#  115-43#  115-43#  115-43#  115-43#  115-43#  115-43#  117-33
        117-33   117-33#  117-33#  118-54   118-54   118-54   118-54   118-54   118-54   118-54   118-54   118-54#  118-54#  118-54#
        118-54#  118-54#  118-55   118-55   118-55   118-55   118-55   118-55   118-55   118-55   118-55   118-55   118-55   118-55
        118-55#  118-55#  118-55#  118-55#  118-55#  118-55#  118-56   118-56   118-56#  118-75   118-75   118-75   118-75   118-75
        118-75   118-75   118-75   118-75   118-75   118-75#  118-75#  118-75#  118-75#  118-75#  118-75#  118-76   118-76   118-76#
        118-77   118-91   118-91   118-91   118-91   118-91   118-91   118-91#  118-91#  118-91#  118-91#  118-92   118-92   118-92
        118-92   118-92   118-92   118-92#  118-92#  118-92#  118-93   119-48   119-48   119-48#  119-48#  122-20   122-20   122-20#
        122-20#  123-21   123-21   123-21#  123-21#  129-133  129-133  129-133# 129-133# 132-22   132-22   132-22#  132-22#  133-21
        133-21   133-21#  133-21#  133-23   133-23   133-23#  133-23#  133-30   133-30   133-30#  133-30#  133-47   133-47   133-47#
        133-47#  133-48   133-48   133-48#  133-48#  133-53   133-53   133-53   133-53   133-53   133-53#  133-53#  133-53#  133-53#
        133-53#  133-65   133-65   133-65#  133-65#  133-66   133-66   133-66#  133-66#  133-67   133-67   133-67#  133-67#  133-69
        133-69   133-69   133-69   133-69   133-69   133-69#  133-69#  133-69#  133-69#  136-23#  136-30#  137-18   137-18   137-18#
        137-18#  137-36   137-36   137-36   137-36   137-36   137-36   137-36#  137-36#  137-36#  137-36#  137-42   137-42   137-42
        137-42   137-42   137-42   137-42#  137-42#  137-42#  137-42#  138-45   138-45   138-45   138-45   138-45   138-45   138-45#
        138-45#  138-45#  138-45#  139-29   139-29   139-29   139-29   139-29#  139-29#  139-29#  141-155  141-155  141-155# 141-155#
        143-54   143-54   143-54#  143-54#
NXTLEV  10-34#   61-99    61-102   106-46   107-23   107-56   107-58   118-60
NXTRCT  110-70   113-2#
NXTTRK  75-12    76-31    77-32    79-2#
OCNT$   44-29#   44-30    44-30    44-30#   44-31    44-31    44-31#   44-32    44-32    44-32#   44-33    44-33    44-33#   44-34
        44-34    44-34#   44-35    44-35    44-35#   44-36    44-36    44-36#   44-37    44-37    44-37#   44-38    44-38    44-38#
        44-39    44-39    44-39#   44-40    44-40    44-40#   44-41    44-41    44-41#   44-42    44-42    44-42#   44-43    44-43
        44-43#   44-44    44-44    44-44#   44-45    44-45    44-45#   44-46    44-46    44-46#   44-47    44-47    44-47#   44-48
        44-48    44-48#   44-49    44-49    44-49#   44-50    44-50    44-50#   44-51    44-51    44-51#   44-52    44-52    44-52#
        44-58    44-58    44-58#   44-59    44-59    44-59#   44-60    44-60    44-60#   44-61    44-61    44-61#   44-62    44-62
        44-62#   44-63    44-63    44-63#   44-64    44-64    44-64#   44-65    44-65    44-65#
ONEPAT  97-78    97-86#
ONEPAX  108-53   108-63#
ONL     24-6     24-34#
ONLYCL  10-49#   58-59    129-82   133-19   133-45   133-63
OTABLE  33-146   34-8     34-11    34-15    34-30    44-30#   44-66    50-8*    50-10
OUTO    74-14    74-16#
OV...    2-39     2-39     2-39#   59-9     59-9     59-9     59-9#    61-2     61-2     61-2     61-2#    62-2     62-2     62-2
        62-2#    71-2     71-2     71-2     71-2#    73-2     73-2     73-2     73-2#    80-2     80-2     80-2     80-2#    84-2
        84-2     84-2     84-2#    93-2     93-2     93-2     93-2#    97-2     97-2     97-2     97-2#    99-2     99-2     99-2
        99-2#    100-2    100-2    100-2    100-2#   103-2    103-2    103-2    103-2#   104-2    104-2    104-2    104-2#   105-2
```

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE S-11
CROSS REFERENCE TABLE (CREF V04.00)

```
        105-2    105-2    105-2#   106-2    106-2    106-2    106-2#   107-2    107-2    107-2    107-2#   108-2    108-2    108-2
        108-2#   109-2    109-2    109-2    109-2#   110-2    110-2    110-2    110-2#   115-2    115-2    115-2    115-2#   118-2
        118-2    118-2    118-2#   119-2    119-2    119-2    119-2#   120-2    120-2    120-2    120-2#   121-2    121-2    121-2
        121-2#   122-2    122-2    122-2    122-2#   123-2    123-2    123-2    123-2#   124-2    124-2    124-2    124-2#   129-2
        129-2    129-2    129-2#   136-2    136-2    136-2    136-2#   141-2    141-2    141-2    141-2#   143-2    143-2    143-2
        143-2#   143-86   143-86#
OVE.AW   99-2#   100-2
OVE.BP   93-2#    97-2
OVE.CC  122-2#   123-2
OVE.CD  108-2#   109-2
OVE.CK  105-2#   106-2
OVE.DA  120-2#   121-2
OVE.EC  106-2#   107-2
OVE.ED  104-2#   105-2
OVE.GS  141-2#   143-2
OVE.ID  143-2#   143-86
OVE.IN  121-2#   122-2
OVE.LM  109-2#   110-2
OVE.MN    2-39#   59-9
OVE.MS   59-9#    61-2
OVE.NL  107-2#   108-2
OVE.NO   62-2#    71-2
OVE.R   100-2#   103-2
OVE.RB   71-2#    73-2
OVE.RC  119-2#   120-2
OVE.RT   80-2#    84-2
OVE.RV  110-2#   115-2
OVE.SO  129-2#   136-2
OVE.S1  136-2#   141-2
OVE.SB   73-2#    80-2
OVE.SC  103-2#   104-2
OVE.SK  115-2#   118-2
OVE.SO  124-2#   129-2
OVE.SP  123-2#   124-2
OVE.ST   84-2#    93-2
OVE.SU   61-2#    62-2
OVE.TS  118-2#   119-2
OVE.W    97-2#    99-2
OVL...     2-39     2-39#    59-9     59-9#    61-2     61-2#    62-2     62-2#    71-2     71-2#    73-2     73-2#    80-2     80-2#
          84-2     84-2#    93-2     93-2#    97-2     97-2#    99-2     99-2#   100-2    100-2#   103-2    103-2#   104-2    104-2#
         105-2    105-2#   106-2    106-2#   107-2    107-2#   108-2    108-2#   109-2    109-2#   110-2    110-2#   115-2    115-2#
         118-2    118-2#   119-2    119-2#   120-2    120-2#   121-2    121-2#   122-2    122-2#   123-2    123-2#   124-2    124-2#
         129-2    129-2#   136-2    136-2#   141-2    141-2#   143-2    143-2#   143-86   143-86#
OVL.AW   44-39   100-2     100-2    100-2#
OVL.BP   44-37    97-2      97-2     97-2#
OVL.CC   44-59   123-2     123-2    123-2#
OVL.CD   44-46   109-2     109-2    109-2#
OVL.CK   44-43   106-2     106-2    106-2#
OVL.DA   44-52   121-2     121-2    121-2#
OVL.EC   44-44   107-2     107-2    107-2#
OVL.ED   44-42   105-2     105-2    105-2#
OVL.GS   44-64   143-2     143-2    143-2#
OVL.ID   44-65   143-86    143-86   143-86#
OVL.IN   44-58   122-2     122-2    122-2#
OVL.LM   44-47   110-2     110-2    110-2#
OVL.MN     2-39     2-39     59-9     59-9     59-9#
```

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE S-12
CROSS REFERENCE TABLE (CREF V04.00)

```
OVL.MS   44-30    61-2     61-2     61-2#
OVL.NL   44-45   108-2    108-2    108-2#
OVL.NO   44-32    71-2     71-2     71-2#
OVL.R    44-40   103-2    103-2    103-2#
OVL.RB   44-33    73-2     73-2     73-2#
OVL.RC   44-51   120-2    120-2    120-2#
OVL.RT   44-35    84-2     84-2     84-2#
OVL.RV   44-48   115-2    115-2    115-2#
OVL.S0   44-62   136-2    136-2    136-2#
OVL.S1   44-63   141-2    141-2    141-2#
OVL.SB   44-34    80-2     80-2     80-2#
OVL.SC   44-41   104-2    104-2    104-2#
OVL.SK   44-49   118-2    118-2    118-2#
OVL.SO   44-61   129-2    129-2    129-2#
OVL.SP   44-60   124-2    124-2    124-2#
OVL.ST   44-36    93-2     93-2     93-2#
OVL.SU   44-31    62-2     62-2     62-2#
OVL.TS   44-50   119-2    119-2    119-2#
OVL.W    44-38    99-2     99-2     99-2#
OVS.AW   99-2#
OVS.BP   93-2#
OVS.CC  122-2#
OVS.CD  108-2#
OVS.CK  105-2#
OVS.DA  120-2#
OVS.EC  106-2#
OVS.ED  104-2#
OVS.GS  141-2#
OVS.ID  143-2#
OVS.IN  121-2#
OVS.LM  109-2#
OVS.MN    2-39#
OVS.MS   59-9#
OVS.NL  107-2#
OVS.NO   62-2#
OVS.R   100-2#
OVS.RB   71-2#
OVS.RC  119-2#
OVS.RT   80-2#
OVS.RV  110-2#
OVS.S0  129-2#
OVS.S1  136-2#
OVS.SB   73-2#
OVS.SC  103-2#
OVS.SK  115-2#
OVS.SO  124-2#
OVS.SP  123-2#
OVS.ST   84-2#
OVS.SU   61-2#
OVS.TS  118-2#
OVS.W    97-2#
OVSTRT    5-10#   50-6     50-7
PARTO    44-8#    44-30    44-30    44-31    44-31    44-31    44-32    44-33    44-34    44-35    44-36    44-37    44-37    44-37
         44-38    44-38    44-39    44-39    44-40    44-40    44-41    44-41    44-42    44-42    44-43    44-43    44-44    44-44
         44-45    44-45    44-46    44-46    44-47    44-47    44-48    44-48    44-49    44-50    44-51    44-52    44-52    44-52
         44-58    44-59    44-60    44-61    44-62    44-63    44-64    44-65
```

CROSS REFERENCE TABLE (CREF V04.00)

```
PART1   44-10#   44-32
PART2   44-12#   44-33    44-34    44-35    44-36
PART3   44-14#   44-31    44-37    44-49    44-50    44-51    44-52
PARTI   44-16#   44-58    44-59    44-60    44-61    44-62    44-63    44-64    44-65
PAT0    30-2     30-21#   50-29
PAT1    30-3     30-39#
PAT10   30-12    30-126#
PAT11   30-13    30-144#
PAT12   30-14    30-147#
PAT13   30-15    30-165#
PAT14   30-16    30-183#
PAT15   30-17    30-188#
PAT2    30-4     30-42#
PAT3    30-5     30-45#
PAT4    30-6     30-48#
PAT5    30-7     30-66#
PAT6    30-8     30-84#
PAT7    30-9     30-102#
PAT8    30-10    30-105#
PAT9    30-11    30-123#
PATEXT  65-7     65-16#
PATPTR  30-2#    97-72    108-46
PHYSA    2-39     2-39     2-39     2-39     2-39#
PLOAD   34-34    34-38#
PNUM    29-20#   108-28*  108-30*  108-44   108-105
PREVBE  73-59    76-2#
PTABLE  33-148   34-10    34-29*   34-36*   34-39    44-7#
RANDOM  64-12    65-12    66-14    67-14    69-2#    71-27    72-16    72-56    80-17    82-22    83-11
RBLOCK  100-52   101-2#
RBNBN    10-22#   25-9     28-22    33-14    61-76    63-14    93-18    95-34    97-37    99-56    99-96    99-108   100-40   103-64
        103-75   109-31   112-25   116-8
RBNFLG   28-8*    28-24*   29-7#    141-41*
RBNTRK    7-30#   27-11    28-10    28-25    28-41    129-66   141-36
RBNTXT   59-278#  97-34    99-103   99-143   100-37   103-70   103-121  104-46   104-80   105-38   105-70   107-32   108-86   115-43
        118-55
RBUFLN    5-69#   46-18    95-20
RCBREQ   10-36#   39-22    43-44    115-32   119-51
RCONT     5-41#
RCTCPS    7-8#    122-59
RCTCSZ    7-39#   111-23   113-12
RCTL$    59-288#  110-30
RCV       6-8#    23-45    38-68    51-60
RCVERR    8-27#   18-35
RCVRDY    8-22#   18-18    23-38    38-34    38-51    51-27
RDSTAT   18-17    18-25#   51-13    51-24    52-13
READ     44-40    95-67    100-10#  103-10
RECALB   44-51    115-34   119-10#  120-10
RECOVR   33-43    39-14    40-2#    141-53
REDWRT   10-23#   61-65    61-80    63-14    64-15    65-6     66-6     67-8     94-9     95-7     95-61
RETRY    10-37#   39-20    39-26    43-26
RETS      7-7#    122-28
REVCT    44-48    99-98    100-48   103-66   103-81   104-67   105-91   107-39   110-10#  115-10
REVEC    10-21#   33-14    63-14    94-11    95-9     95-63    100-34   100-46   103-62   104-29   104-65   104-86   105-80   105-89
        107-37   110-20   110-28   114-7
REVS      7-16#
REVSOK  110-63   111-2#
```

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

```
START   17-9     46-1#
STATUS   6-10#   18-32
STOP    31-48    31-52#   34-27    46-12    54-41
STSERR 118-33   118-65#
STSEXT 118-29   118-64   118-71   118-78   118-86   118-90   118-95#
STSRES   8-47#   24-8
SUBTRV 130-3    130-13#
SUBUNT  44-87#  129-34*
SUNIT  141-16*  141-156  141-160  141-171* 141-172* 142-2#
SUSETL  55-53#   58-121
SWAPBE 125-23   126-2#
T1MSIZ   8-4#
T2CMD    8-6#
T2DLL    8-5#
T4BB1    8-9#    57-8     58-78
T4BB2    8-10#   58-90
T4MPRM   8-7#    50-24
T4MXFR   8-13#   37-1     97-108   109-39
T4SEEK   8-12#  118-45
T4SOFT   8-11#   19-7     23-106    39-13    98-43    99-116  101-22   103-94   104-63   105-61   115-51   118-53   118-74
T4UPRM   8-8#    56-7     58-8     141-168
TALK    23-2#    31-41    40-10     40-16    40-53    40-63   107-50   117-10   119-23   122-17   123-16   129-37   141-57   141-64
TG$    138-33*  138-36*  138-45   139-29   139-91#
TGS     45-23#  138-12   138-38   139-5    139-45
THREBE  71-31    71-34#
TLEN.U   9-78#    9-79    55-22
TLKHST  19-2#    23-50    93-15
TMEMPL  93-25*   96-6*    96-7     96-10#
TO     122-44   122-48   122-69#
TRACK   29-10#   95-6     141-101
TRACKS  10-53#   58-14    82-8     84-20    84-26    84-37   138-30   139-40   140-11
TRAV   129-48   130-2#   130-9
TRK$    59-280# 138-36
TRKGRP   7-27#   82-11    84-41    91-7     134-8    138-35  139-44
TROOT   46-5     49-2#
TRYAGN  71-27#   71-35
TSTNEC 115-52   116-2#
U.CBN    9-41#    9-42    61-40    61-42*   61-44    61-46*  61-68*   61-70*   68-2*    68-4*    93-23    93-24    97-33    97-33
        99-29    99-30    99-73    99-74*   99-76    99-78*  100-36   100-36   102-7    102-8    103-39   103-40*  103-42   103-44*
       106-29   106-30*  106-32   106-34*  107-31   107-31  110-43*  110-48*  111-17   111-17   111-24   111-25*  111-27   111-29*
       113-7    113-8    113-20*  113-21*  113-41   113-41  114-13*  114-17*  115-42   115-42   116-13   116-14   118-54   118-54
U.CCNT   9-20#    9-21    85-12    85-14*   85-16    85-23*  86-10    86-12*   86-14    86-21*   89-8*    89-10*   90-11*   90-13*
U.CCOP   9-44#    9-45   110-38*  111-6    111-21*  113-9   141-46*  141-129  141-131*
U.CCYL   9-48#    9-49    93-29    93-31    98-15    98-15   99-105   99-105   99-145   99-145   101-11   101-11   103-72   103-72
       103-123  103-123  104-48   104-49   104-82   104-83  105-40   105-41   105-65   105-65   105-72   105-73   107-33   107-33
       108-88   108-88   116-21   117-13   118-55   118-56  118-75   118-76   118-92   118-92   119-55*  119-56*  121-19*  121-20*
U.CGRP   9-49#    9-50    93-33    98-15    99-105   99-145  101-11   103-72   103-123  104-48   104-82   105-40   105-65   105-72
       107-32   108-88   118-55   118-75   118-92   119-57* 121-21*
U.COPY   9-43#    9-44   111-8    122-63*  141-42   141-132
U.CSEC   9-26#    9-27    33-20*   33-68*   61-26    61-39*  61-72*   61-74    68-6*    99-70    99-71*   103-33   103-34*  106-19
       106-20*  110-54   114-11
U.CTRK   9-22#    9-23    87-14    87-17*   88-11    88-14*  91-9*
U.ECCT   9-32#    9-33   105-31   122-41*  141-120
U.ELEV   9-29#    9-30    61-91*   104-44   104-60   104-78  105-36   105-52   105-68   107-25   107-28   107-61   107-63*
U.LCYL   9-50#    9-51    99-106   99-106   99-146   99-146  103-73   103-73   103-124  103-124  117-17   118-75   118-75   118-91
       118-91
```

CROSS REFERENCE TABLE (CREF V04.00)

CROSS REFERENCE TABLE (CREF V04.00)

```
UMASK    45-17#   55-7*    55-43    55-45*
UNITS    51-10    53-7     53-19    53-26    54-7     54-36    54-53#   55-8     55-47
UNSSUC    8-44#   23-85
UP       73-17    73-70#
UPG      84-30    87-2#
UPT      84-28    85-2#    87-39
UREAD     6-14#   34-16
UTOTST    8-14#   53-4
UWRITE    6-15#
VALBIT   58-14#   58-16
VALID     6-39#   23-84    23-87    38-10    38-47
WAITSI    6-13#   98-12   101-8    141-106
WBLOCK   97-98    98-2#
WBUFLN    5-68#    5-69    46-24    94-13
WCHECK   10-54#   33-14    58-14    61-49    61-65    63-14    66-8     66-17    67-10
WCHKAL   10-56#   58-14    66-12
WCONT     5-39#
WONLY    10-47#   58-14    64-8    136-21
WREAL     5-43#   95-11
WRITBT   24-23#   40-61*
WRITE    44-38    95-65    97-10#   99-10
WSTOP     5-38#   95-13    99-42
XBNCYL    7-40#  129-91
XFERRT    7-5#
XOPLP0  108-50#  108-62
XOPLP1  108-54#  108-61
XOPLP2  108-64#  108-68
XREAD     6-5#   101-15   141-109
XWRITE    6-6#    98-20
ZEREDC   22-13     22-15#
```

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE M-1
CROSS REFERENCE TABLE (CREF V04.00)

UDAT4 DISK EXERCISER DMACR X04.01 13-APR-82 15:49:22 PAGE M-2
CROSS REFERENCE TABLE (CREF V04.00)

```
        105-65  105-65  105-65  105-65  105-65  105-65  105-68  105-68  105-68  105-68  105-69  105-69  105-69  105-70
        105-70  105-71  105-71  105-71  105-72  105-72  105-72  105-73  105-73  105-73  107-31  107-31  107-31  107-31
        107-32  107-32  107-32  107-32  107-32  107-33  107-33  107-53  108-80  108-81  108-84  108-85  108-85  108-85
        108-86  108-86  108-87  108-87  108-87  108-88  108-88  108-88  108-88  108-105 108-105 108-105 108-105 108-105
        108-105 108-105 108-105 108-105 108-105 108-105 108-106 108-106 108-106 108-106 111-10  111-14  111-16  111-17
        111-17  111-17  111-17  112-44  112-48  112-50  112-51  112-51  113-34  113-38  113-40  113-41  113-41  113-41
        113-41  115-42  115-42  115-42  115-42  115-43  115-43  115-43  115-43  115-43  115-43  115-43  117-33  118-54
        118-54  118-54  118-54  118-54  118-55  118-55  118-55  118-55  118-55  118-55  118-56  118-75  118-75  118-75
        118-75  118-75  118-75  118-76  118-77  118-91  118-91  118-91  118-91  118-92  118-92  118-92  118-93  119-48
        122-20  123-21  129-133 132-22  132-22  133-21  133-21  133-23  133-23  133-30  133-30  133-47  133-48  133-53
        133-53  133-53  133-53  133-65  133-66  133-67  133-67  133-69  133-69  133-69  133-69  136-23  136-30  137-18
        137-18  137-36  137-36  137-36  137-36  137-42  137-42  137-42  137-42  138-45  138-45  138-45  138-45  139-29
        139-29  139-29  141-155 141-155 143-54  143-54
MSG      12-20#  24-6    24-8    24-9    24-10   24-11   24-12   24-13   24-15   44-74   44-75   44-76
MSSG     15-138# 33-77   105-65  136-30
NDERR    15-124# 18-12   23-73   23-77   36-21   36-29   40-35   97-35   98-16   98-37   99-107  99-139  99-147  100-38
        101-12  103-74  103-117 103-125 104-50  104-84  105-42  105-74  107-34  108-107 111-18  112-52  115-44  118-57
        118-77  118-93  143-55
OVTERM   2-39    2-39#   2-39#   59-9    59-9#   61-2    61-2#   62-2    62-2#   71-2    71-2#   73-2    73-2#   80-2
         80-2#   84-2    84-2#   93-2    93-2#   97-2    97-2#   99-2    99-2#   100-2   100-2#  103-2   103-2#  104-2
        104-2#  105-2   105-2#  106-2   106-2#  107-2   107-2#  108-2   108-2#  109-2   109-2#  110-2   110-2#  115-2
        115-2#  118-2   118-2#  119-2   119-2#  120-2   120-2#  121-2   121-2#  122-2   122-2#  123-2   123-2#  124-2
        124-2#  129-2   129-2#  136-2   136-2#  141-2   141-2#  143-2   143-2#  143-86
POP      14-11#  18-42   19-7    20-23   22-17   23-21   23-26   23-51   23-89   23-100  23-106  23-107  25-25   33-77
        33-137  34-19   34-28   34-38   38-73   38-80   39-13   39-17   39-31   39-51   51-47   51-67   51-70   52-4
        52-17   52-33   53-15   53-24   54-28   54-34   55-41   57-10   58-10   58-80   58-92   58-109  58-116  69-26
        70-15   72-34   72-41   72-63   73-100  78-13   79-38   79-42   79-61   81-22   82-26   90-30   95-23   97-92
        98-18   98-21   98-43   99-49   99-116  101-22  102-24  103-94  104-63  105-28  105-34  105-44  105-63  105-65
        108-70  108-73  108-77  115-51  117-23  118-53  118-74  120-44  124-31  124-34  126-23  127-32  129-33  129-38
        129-49  129-56  129-139 130-4   130-21  130-38  130-41  131-50  133-72  135-30  136-30  136-82  137-45  139-84
        139-88  141-170 141-174
PUSH     14-3#   18-30   19-7    20-12   22-7    23-7    23-22   23-31   23-54   23-106  25-7    33-9    33-77   34-6
        34-20   34-32   35-2    36-6    38-14   39-13   51-11   51-44   51-55   52-11   53-10   54-13   54-29   55-29
        55-53   55-67   57-6    58-6    58-76   58-88   69-7    70-7    72-5    72-15   72-55   73-89   78-7    79-8
        79-28   81-4    82-4    82-18   90-7    95-19   97-70   98-6    98-43   99-28   99-116  101-22  102-6   103-94
        104-63  105-23  105-57  105-65  108-22  108-74  115-51  117-16  118-53  118-74  120-19  124-16  124-22  126-7
        127-19  129-16  129-25  129-36  129-47  129-53  130-2   130-19  130-30  131-7   133-8   135-7   136-16  136-30
        137-7   138-11  139-61  141-15  141-25
REPSFT   15-9#   19-7    23-106  39-13   98-43   99-116  101-22  103-94  104-63  115-51  118-53  118-74
RXOR     16-10#  22-11
SOFTER   15-4#   23-24   23-40   23-56   23-60   23-64   23-68   23-72   23-82   36-28   99-86   99-117  99-122  99-126
        99-131  99-136  103-52  103-86  103-90  103-95  103-100 103-104 103-109 103-114 104-44  104-78  105-29  105-35
        105-68  118-54  118-75  118-91
SYSFTL   15-65#
```