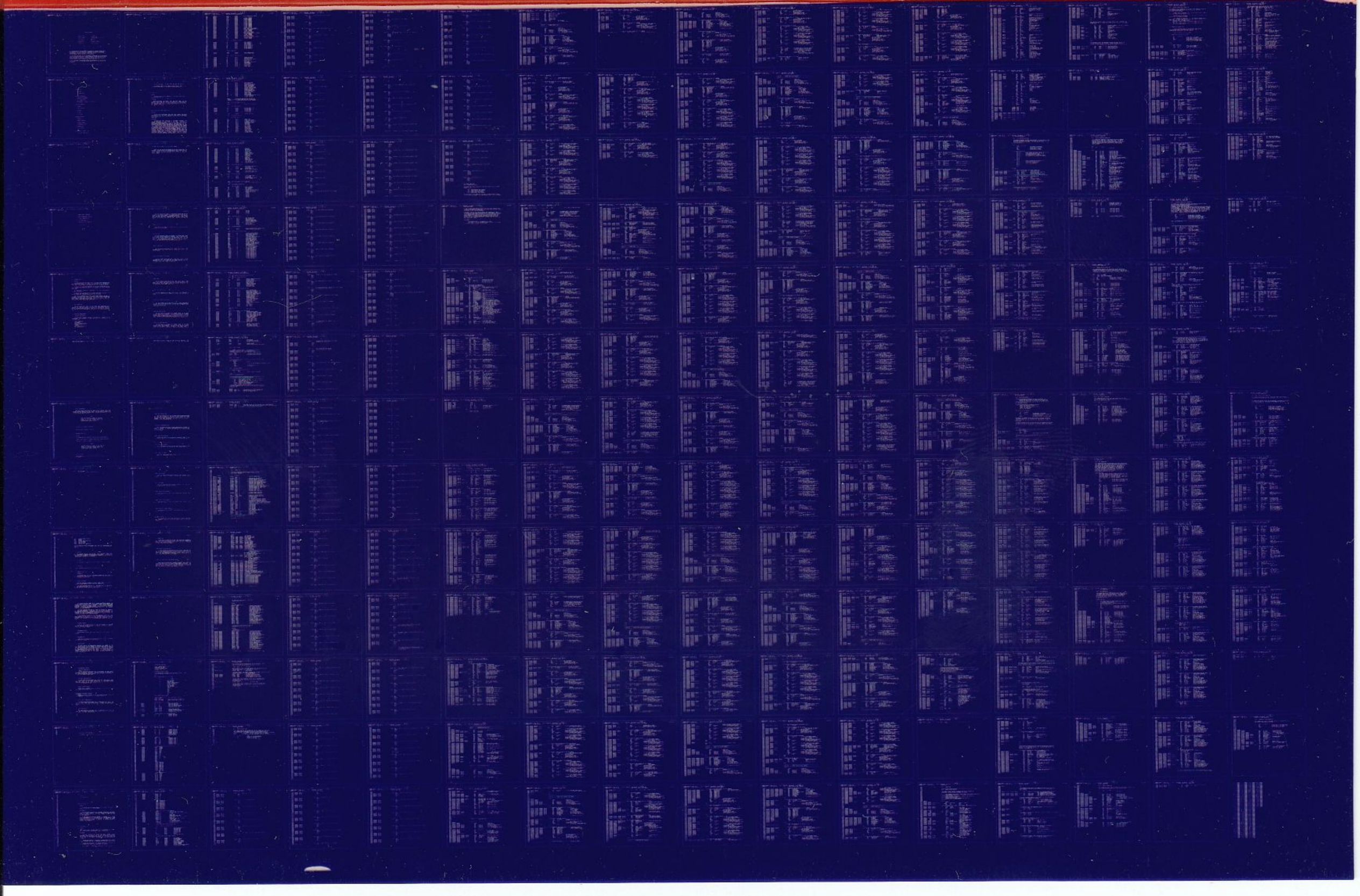
RM02,RM03

FCTNL TST 2
CZRMDC0

AH-B001C-MC

FICHE 1 OF 2

MAR 1979
digital
MADE IN USA



RM02, RM03

FCTNL TST 2
CZRMDC0

AH-B001C-MC

FICHE 2 OF 2

MAR 1979
digital
MADE IN USA

```
1                       .REM    \
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17                                  IDENTIFICATION
18                                  --------------
19
20
21                      PRODUCT CODE:           AC-B000C-MC
22
23                      PRODUCT NAME:           CZRMDC0 RM03/2 FCTNL TST 2
24
25                      DATE CREATED:           FEB 1979
26
27                      MAINTAINER:             DIAGNOSTIC GROUP
28
29                      AUTHOR:                 DOUG RIIKONEN
30
31
32
33
34
35
36
37          THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE
38          AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
39          CORPORATION.  DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY
40          FOR ANY ERRORS THAT MAY APPEAR IN THIS MANUAL.
41
42          THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURSHASER
43          UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED
44          (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH
45          SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.
46
47          DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE
48          OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY
49          DIGITAL.
50
51              COPYRIGHT (C) 1977,1979 DIGITAL EQUIPMENT CORPORATION
```

```
                        .PAGE


                                                CONTENTS


                1.  INTRODUCTION

                    1.  ABSTRACT

                    2.  UNIT UNDER TEST

                2.  OPERATING REQUIREMENTS

                    1.  HARDWARE REQUIREMENTS

                    2.  MEDIA REQUIREMENTS

                    3.  PREREQUISITE DIAGNOSTIC PROGRAMS

                3.  OPERATING PROCEDURE

                    1.  LOADING

                    2.  SWITCH OPTIONS

                    3.  STARTING

                    4.  HALTING

                    5.  RESTARTING

                4.  OPERATOR INTERFACE

                    1.  PROGRAM I.D.

                    2.  CONSOLE DIALOGUE

                    3.  PROGRESS REPORTS

                    4.  PERFORMANCE REPORTS

                    5.  PROGRAM HALTS

                    6.  ERROR REPORTS

                5.  ENVIRONMENTAL SUPPORT

                    1.  PROCESSOR COMPATIBILITY
```

108

2. DUAL PORT CONFIGURATIONS

3. MEMORY PARITY HARDWARE

4. MEMORY MANAGEMENT HARDWARE

5. ACT,APT COMPATIBILITY

6. XXDP COMPATIBILITY

7. OPERATING SYSTEM COMPATIBILITY

6. TEST DESCRIPTION

1.0 INTRODUCTION

1.1 ABSTRACT

THE RM03 SUBSYSTEM FUNCTIONAL TEST IS A STAND ALONE PROGRAM WHICH USES FUNCTIONAL MEANS TO VERIFY THE OPERABILITY OF THE RM03 DISK SUBSYSTEM. IN PARTICULAR, THE PROGRAM SERVES THE FOLLOWING PURPOSES:

TO EXPLICITLY ESTABLISH CONFIDENCE IN THE BASIC OPERATIONS OF THE DISK DRIVE, INCLUDING MECHANICAL POSITIONING AND DATA TRANSFER OPERATIONS;

TO IMPLICITLY ESTABLISH CONFIDENCE IN THE DRIVE/ADAPTER ELECTRICAL INTERFACE;

TO VERIFY THE FUNCTIONALITY OF THE RM03 SUBSYSTEM, INCLUDING THE MASSBUS CONTROLLER, MASSBUS ADAPTER AND THE DISK DRIVE.

THE TEST IS COMPRISED OF 3 PARTS, WHICH WOULD NORMALLY BE RUN IN SEQUENCE, STARTING WITH PART 1. BRIEFLY, PART 1 TESTS HOUSEKEEPING AND MECHANICAL POSITIONING OPERATIONS; PART 2 TESTS WRITE, READ AND WRITE CHECK OPERATIONS USING HEADER AND DATA; PART 3 TESTS WRITE, READ AND WRITE CHECK OPERATIONS USING DATA.

1.2 UNIT UNDER TEST

THE UNIT UNDER TEST (UUT) IS THE RM03 DISK SUBSYSTEM WHICH CONSISTS OF THE RHXX MASSBUS CONTROLLER ,THE RM03 MASSBUS ADAPTER , AND THE STORAGE MODULE DISK DRIVE. NOTE THAT A DISK PACK IS REQUIRED FOR TESTING AND IS CONSIDERED AN INTEGRAL OF THE STORAGE MODULE DISK DRIVE.

2.0 OPERATING REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

THE FOLLOWING MINIMUM HARDWARE CONFIGURATION, ASSUMED TO BE OPERATIONAL, IS REQUIRED TO LOAD AND EXECUTE THE RM03 SUBSYSTEM FUNCTIONAL TEST:

PDP-11 PROCESSOR
16K MEMORY
KW11-L OR KW11-P CLOCK
PROGRAM LOADING DEVICE
TERMINAL
RH11 OR RH70 CONTROLLER
UNIT UNDER TEST,

```
181                     WHERE THE UNIT UNDER TEST CONSISTS OF ONE TO EIGHT RM03 ADAPTERS, DISK
182                     DRIVES AND DISK PACKS.
```

2.2 MEDIA REQUIREMENTS

EACH UNIT BEING TESTED MUST BE LOADED WITH A SCRATCH DISK PACK BEFORE TESTING BEGINS ON THAT UNIT. THE PACK MAY BE FORMATTED OR UNFORMATTED, BUT MUST NOT CONTAIN NEEDED INFORMATION BECAUSE THE PACK WILL BE WRITTEN DURING THE TEST.

NOTE

WHEN THE PROGRAM IS LOADED VIA THE RM03, DRIVE 0 IS NOT TESTED UNLESS THE OPERATOR SPECIFICALLY ENTERS THE DRIVE NUMBER DURING CONSOLE DIALOGUE.

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

RM03 DISKLESS DIAGNOSTIC, CZRMJ-B

3.0 OPERATING PROCEDURE

3.1 LOADING

THE PROGRAM MAY BE LOADED BY EITHER OF THE FOLLOWING MEDIA:

. PAPER TAPE, USING THE STANDARD PAPER TAPE LOADING PROCEDURE;

.XXDP MEDIA, USING THE APPROPRIATE LOADING DEVICE.

NOTE

WHEN THE PROGRAM IS LOADED VIA THE RM03, DRIVE 0 IS NOT TESTED UNLESS THE OPERATOR SPECIFICALLY ENTERS THE DRIVE NUMBER DURING CONSOLE DIALOGUE.

3.2 SWITCH OPTIONS

THE FOLLOWING SWITCH OPTIONS ARE PROVIDED TO ENHANCE THE UTILITY OF THE PROGRAM.

```
                                SW15    HALT ON ERROR
                                SW14    LOOP ON TEST (CURRENTLY BEING EXECUTED)
```

SW13 INHIBIT ERROR TYPEOUTS
SW12 UNUSED
SW11 INHIBIT TEST ITERATIONS
SW10 BELL ON ERROR
SW09 LOOP ON ERROR
SW08 LOOP ON TEST IN SW07-00

THE LOW ORDER 8 SWITCHES (SW07-SW00), ARE USED IN CONJUNCTION WITH SW08 TO SPECIFY THE OCTAL NUMBER OF THE TEST WHICH THE PROGRAM WILL LOOP ON.

3.3 STARTING

THE PROGRAM STARTS AT LOCATION 200, WHICH PROVIDES WORST CASE TEST CONDITIONS IF RUNNING IN AN AUTOMATIC ENVIRONMENT. IF RUNNING IN A STAND-ALONE ENVIRONMENT, THE PROGRAM USES CONSOLE DIALOGUE TO ALLOW THE OPERATOR TO CONTROL TEST CONDITIONS.

3.4 HALTING

THE PROGRAM CAN BE HALTED BY TYPING CONTROL C FROM THE CONSOLE.

3.5 RESTARTING

THE PROGRAM CAN BE RESTARTED AT ADDRESS 200.

4.0 OPERATOR INTERFACE

4.1 PROGRAM ID

THE PROGRAM TYPES ITS TITLE AND MAINDEC NUMBER THE FIRST TIME IT IS STARTED AFTER BEING LOADED. PROGRAM IDENTIFICATION DOES NOT OCCUR IF THE PROGRAM IS RESTARTED.

4.2 CONSOLE DIALOGUE

WHEN THE PROGRAM IS RUNNING IN STAND ALONE MODE, IT ENTERS A CONSOLE DIALOGUE SEQNENCE AFTER TYPING THE PROGRAM I.D..

THE FIRST QUESTION TYPED OUT IS: "TYPE HELP TEXT (Y OR N)??". IF THE OPERATOR RESPONDS WITH A Y, THE PROGRAM WILL TYPE A BRIEF HELP MESSAGE WHCIH WILL LIST SWITCH OPTIONS, ETC.

THE SECOND QUESTION TYPED OUT IS, "CHANGE RM03 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N)??". IF THE UNIBUS ADDRESS OF THE RM03 IS NON STANDARD, THE OPERATOR SHOULD RESPOND Y, THEN ANSWER SUBSEQUENT QUESTIONS TO SPECIFY THE UNIBUS ADDRESS, VECTOR ADDRESS AND INTERRUPT PRIORITY. IF THE OPERATOR DOES NOT RESPOND WITH A Y, THE PROGRAM SKIPS TO THE THIRD QUESTION.

THE THIRD QUESTION TYPED OUT IS, "TYPE (A) TO TEST ALL DEVICES, OR TYPE DEVICE NUMBER(S). TERMINATE INPUT WITH CARRIAGE RETURN". IF THE OPERATOR TYPES A, ALL POSSIBLE DEVICES ARE TESTED. OTHERWISE, THE OEPRATOR CAN TYPE THE NUMBER(S) OF THE DEVICE(S) HE WANTS TESTED, AND TERMINATE HIS INPUT WITH A CARRIAGE RETURN.

IF THE PROGRAM IS RESTARTED, THE FIRST QUESTION TYPED IS, "USE SAME DEVICES (Y OR N)??". IF THE OPERATOR TYPES Y, THE TEST IS RESTARTED USING THE SAME DEVICES AS THE LAST TIME, OTHERWISE, THE TEST RESTARTS THE DIALOGUE AS IF THE PROGRAM WERE STARTED FOR THE FIRST TIME.

4.3 PROGRESS REPORTS

AN END OF PASS REPORT OCCURS EACH TIME THE PROGRAM IS EXECUTED FOR ALL DEVICES IN THE TEST QUE. THE END OF PASS REPORT INCLUDES A MESSAGE AND AN ERROR SUMMARY.

4.4 PERFORMANCE REPORT

NO PERFORMANCE REPORTS ARE GIVEN DURING THE EXECUTION OF THE PROGRAM.

4.5 PROGRAM HALTS

THERE ARE NO SCHEDULED HALTS DURING THE EXECUTION OF THE PROGRAM. PROCESSUR HALTS ARE DUE TO THE TRAP CATCHER.

4.6 ERROR REPORTS

THE FIRST LINE OF THE ERROR REPORT CONTAINS THE NUMBER OF THE UNIT BEING TESTED, THE TEST NUMBER, THE ERROR NUMBER AND THE VALUE OF THE PROGRAM COUNTER WHERE THE ERROR WAS CALLED. THIS LINE IS FOLLOWED BY THE ERROR MESSAGE: ONE OR MORE LINES OF TEXT WHICH GIVE A BRIEF, YET COMPREHENSIVE DESCRIPTION OF THE ERROR. THE ERROR MESSAGE IS NORMALLY FOLLOWED BY ONE OR MORE PAIRS OF LINES CONTAINING DATA HEADERS AND DATA PERTININENT TO THE ERROR, INCLUDING EXPECTED AND ACTUAL TEST RESULTS.

5.0 ENVIRONMENTAL SUPPORT

5.1 PROCESSOR COMPATIBILITY

THE RM03 SUBSYSTEM FUNCTIONAL TEST IS EXECUTABLE ON ANY PDP-11 PROCESSOR, PROVIDING PREVIOUSLY MENTIONED HARDWARE REQUIREMENTS ARE MET, AND PROVIDING THAT DATA THROUGHPUT ON THE SYSTEM IS SUFFICIENT TO SUSTAIN DATA TRANSFER OPERATIONS.

5.2 DUAL PORT CONFIGURATIONS

THE RM03 SUBSYSTEM FUNCTIONAL TEST DOES NOT SPECIFICALLY TEST DUAL PORT LOGIC IN THE RM03 ADAPTER BUT IS EXECUTABLE ON RM03 SUBSYSTEMS HAVING THE DUAL PORT OPTION PROVIDING THE DUAL PORT SWITCH IS SET TO THE APPROPRIATE PORT (A OR B).

5.3 MEMORY PARITY HARDWARE

MEMORY PARITY HARDWARE IS NOT USED DURING THE EXECUTION OF THE RM03 SUSBYSTEM FUNCTIONAL TEST.

5.4 MEMORY MANAGEMENT HARDWARE

MEMORY MANAGEMENT HARDWARE IS NOT USED DURING THE RM03 SUSBYSTEM FUNCTIONAL TEST. CAPABILITIES OF THE MASSBUS CONTROLLER.

5.5 ACT11, APT11 COMPATIBILITY

THE RM03 SUSBYSTEM FUNCTIONAL TEST IS COMPATIBLE WITH ACT11 AND APT11 IN BOTH DUMP AND AUTOMATIC MODES. FURTHER, THE PROGRAM WILL EXECUTE A QUICK PASS DURING THE FIRST PASS IN SUPPORT OF QUICK VERIFY MODE.

5.6 XXDP COMPATIBILITY

THE RM03 SUSBYSTEM FUNCTIONAL TEST IS COMPATIBLE WITH XXDP IN DUMP AND CHAIN MODES, AND PROVIDES MEDIA PROTECTION IN THE CASE WHERE THE RM03 IS THE XXDP LOADING DEVICE.

5.7 OPERATING SYSTEM COMPATIBILITY

THE PROGRAM IS NOT COMPATIBLE WITH ANY SOFTWARE OPERATING SYSTEM.

6.0 TEST DESCRIPTION

CONTROLLER ACCESS TEST

PURPOSE:

TO VERIFY THAT THE UNIBUS ADDRESS OF THE RM03 SUBSYSTEM IS CORRECT, AS DEFINED AT LOCATION $BASE.

PROCEDURE:

THE TEST TRIES TO ACCESS ALL MASSBUS CONTROLLER REGISTERS USING THE $BASE ADDRESS. REGISTER CONTENTS ARE IGNORED DURING THE TEST, AND THE TEST FAILS IF A BUS TIMEOUT OCCURS FOR ANY REGISTER TRANSFER.

IF THE TEST FAILS AND THE PROGRAM IS RUNNING IN A STAND ALONE ENVIRONMENT, I.E., LOCATION 42 IS 0, THE PROGRAM WILL JUMP TO LOCATION 204 WHICH ALLOWS THE OPERATOR TO CHANGE THE $BASE ADDRESS VIA CONSOLE DIALOGUE. OTHERWISE, THE PROGRAM ESCAPES TO THE END OF PASS HANDLER.

DEVICE AVAILABLE TEST

PURPOSE:

TO VERIFY THAT THE UNIT UNDER TEST IS AVAILABLE FOR TESTING, AND IS NOT LOCKED OR PROGRAMMED TO THE ALTERNATE PORT.

PROCEDURE:

THIS TEST SELECTS THE DEVICE AND READS CONTROL STATUS REGISTERS 1 AND 2 TO VERIFY THAT THE SELECTED DEVICE IS AVAILABLE FOR TESTING, AS INDICATED BY DVA STATUS, BIT 11 OF RMCS1 AND NED STATUS, BIT 12 OF RMCS2. THE RESULTS OF THE TEST CAN VARY AS FOLLOWS:

.NONEXISTANT DEVICE - THE DEVICE IS NONEXISTENT OR IS LOCKED ON THE ALTERNATE PORT AND IS THEREFORE NOT AVAILABLE FOR TESTING;

.DEVICE NOT AVAILABLE - THE DEVICE EXISTS BUT IS SEIZED BY THE ALTERNATE PORT AND IS NOT AVAILABLE FOR TESTING;

.BUS TIMEOUT - THE MASSBUS CONTROLLER FAILED TO DETECT A

```
   466                                  NONEXISTENT DEVICE;
```

.DEVICE AVAILABLE - THE DEVICE IS AVAILABLE FOR TESTING.

THE PROGRAM JUMPS TO THE SUBPASS HANDLER WHICH SELECTS THE NEXT DRIVE TO BE TESTED IF THE DEVICE IS NOT AVAILABLE.

DRIVE TYPE TEST

PURPOSE:

TO VERIFY THAT THE UNIT UNDER TEST IS AN RM03 SINGLE PORT OR DUAL PORT SUBSYSTEM.

PROCEDURE:

THIS TEST READS THE DRIVE TYPE REGISTER, RMDT, OF THE SELECTED DEVICE AND VERIFIES THAT THE DEVICE IS A SINGLE PORT OR DUAL PORT RM03 SUBSYSTEM. IF THE SELECTED DEVICE IS NOT AN RM03, THE PROGRAM JUMPS TO THE SUBPASS HANDLER WHICH WILL SELECT THE NEXT DEVICE FOR TESTING.

WRITE/READ HEADER AND DATA (FORMAT) TESTS

PURPOSE:

TO TEST WRITE HEADER AND DATA AND READ HEADER AND DATA FUNCTIONALITY OF THE RM03 SUBSYSTEM USING A SET OF VARIABLES WHICH INCLUDE WORD COUNT, HEAD MOTION, HEAD SWITCHING AND ERROR CONDITIONS.

PROCEDURE:

ALTHOUGH EACH TEST EXERCISES A DIFFERENT VARIABLE, THE GENERAL PROCEDURE OF EACH TEST IS THE SAME. THE DRIVE IS INITIALIZED AND RECALIBRATED IF "PIP" OR "SKI" ARE ACTIVE SO THAT THERE ARE NO ERRORS WHEN A TEST BEGINS. FOLLOWING THAT, THE TEST PERFORMS ANY EXPLICIT SEEKS REQUIRED FOR THE CONDITIONS OF THE TEST. REGISTERS ARE PRESET AND THE WRITE HEADER AND DATA COMMAND IS EXECUTED. WHEN THE WRITE COMMAND IS COMPLETE, THE TEST STORES ALL SUBSYSTEM STATUS AND CHECKS FOR PRIMARY ERRORS WHICH PRECLUDE OTHER STATUS CHECKS. IF THERE ARE NO PRIMARY ERRORS, THE TEST VERIFIES THE RESULTS OF THE WRITE COMMAND AND THEN CHECKS FOR SECONDARY ERRORS. LOOP ADDRESSES ARE MODIFIED FOLLOWING THE SUCCESSFUL COMPLETION OF THE WRITE COMMAND IN ORDER TO SHORTEN EXECUTION TIMES AND ENHANCE SCOPING LOOPS, THEN THE PROGRAM EXECUTES THE READ HEADER AND DATA PORTION OF THE TEST, VERIFYING

THE SAME TYPE OF ERRORS AS IN THE WRITE COMMAND.

NOTE THAT THE SECTOR USED DURING A TEST MAY DIFFER FROM THE PROGRAM LISTING BECAUSE THE PROGRAM SUBSTITUTES A GOOD SECTOR IF THE ONE SELECTED IS LISTED IN THE BAD BLOCK TABLE OF THE LAST TRACK. UNLESS SPECIFIED OTHERWISE, ALL TESTS ARE IN 16 BIT (32 SECTOR) FORMAT.

FORMAT ZEROS - 18

THE TEST SEEKS TO CYLINDER 0, THEN WRITES HEADER AND DATA ON SECTOR 0 IN 18 BIT FORMAT. THE HEADER AND DATA FIELDS ARE ALL ZEROS, CAUSING THE DEVICE TO USE NORMAL WRITE GATE. THE HEADER AND DATA ARE READ AND COMPARED WITH THE WRITE BUFFER. THE INITIAL SEEK POSITIONS THE HEAD SUCH THAT THERE IS NO IMPLIED SEEK.

FORMAT ZEROS - 16

THIS TEST IS THE SAME AS THE PREVIOUS TEST, EXCEPT THAT DATA IS WRITTEN IN 16 BIT FORMAT.

ZERO FILL TEST

THE TEST EXECUTES A SEEK TO CYLINDER 0 TO INSURE THAT THERE IS NO HEAD MOTION DURING DATA TRANSFER. THIS IS FOLLOWED BY A WRITE HEADER AND DATA COMMAND WITH THE WORD COUNT EQUAL TO THE SIZE OF THE HEADER WHICH CAUSES THE RH70 TO ZERO FILL THE DATA FIELD. THE READ HEADER AND DATA COMMAND THAT FOLLOWS READS A FULL SECTOR AND VERIFIES THAT DATA WAS ZERO FILLED.

FORMAT CHECK ZEROS

THE TEST WRITES HEADER AND AN ALL ZEROS DATA FIELD, THEN PERFORMS A WRITE CHECK HEADER AND DATA COMMAND AND VERIFIES THERE ARE NO ERRORS.

FORMAT CHECK ZEROS W/ WCE ERROR

THE TEST WRITES HEADER AND AN ALL ZEROS DATA FIELD. AFTER COMPLEMENTING THE LAST WORD OF THE WRITE BUFFER, THE TEST PERFORMS A WRITE CHECK HEADER AND DATA COMMAND AND VERIFIES THAT THE CORRECT WRITE CHECK ERROR IS DETECTED.

FORMAT ONES

THE TEST WRITES HEADER AND AN ALL ONES DATA FIELD, THEN READS THE HEADER AND DATA, VERIFYING THE READ BUFFER WITH THE WRITE BUFFER. THE ALL ONES FIELD IS A CONSTANT FREQUENCY, AND THE DRIVE SHOULD USE NORMAL WRITE GATE.

FORMAT CHECK ONES

THE TEST FORMATS AN ALL ONES DATA FIELD, THE PERFORMS A WRITE CHECK HEADER AND DATA COMMAND, VERIFYING THAT THERE ARE NO ERRORS.

FORMAT CHECK ONES W/ WCE ERROR

THE TEST FORMATS AN ALL ONES DATA FIELD, THEN COMPLEMENTS THE LAST WORD OF THE WRITE BUFFER. A WRITE CHECK HEADER AND DATA COMMAND IS EXECUTED, AND THE TEST VERIFIES THAT THE CORRECT WRITE CHECK ERROR IS DETECTED.

FORMAT MULTIPLE SECTORS

THE TEST SEEKS TO CYLINDER 0 TO INSURE THERE IS NO HEAD MOTION DURING DATA TRANSFER. THE WRITE HEADER AND DATA COMMAND FOLLOWS, WITH THE WORD COUNT EQUAL TO MULTIPLE SECTORS. THE WORD COUNT DURING THE READ HEADER AND DATA PORTION OF THE TEST IS SET FOR ALL OF THE FIRST SECTOR AND THE HEADER OF THE SECOND SECTOR.

FORMAT WITH HEAD SWITCHING

THE TEST SEEKS TO CYLINDER 0 TO INSURE THERE IS NO HEAD MOTION DURING DATA TRANSFER. THE WRITE HEADER AND DATA COMMAND STARTS WITH CYLINDER 0, TRACK 0, SECTOR 31. THE WORD COUNT IS EQUAL TO MULTIPLE SECTORS WHICH CAUSES THE SUBSYSTEM TO SWITCH

```
                                        FROM TRACK 0 TO TRACK 1 AFTER THE FIRST SECTOR IS  WRITTEN.   THE
                                        READ  HEADER  AND  DATA  COMMAND  USES  THE  SAME  WORD COUNT AND
```

PAGE 31

STARTING SECTOR.

FORMAT WITH IMPLIED SEEK

THIS TEST SEEKS TO THE LAST CYLINDER PRIOR TO WRITING HEADER AND DATA ON CYLINDER 0. THE EXPLICIT SEEK INSURES THAT THERE WILL BE MAXIMUM HEAD MOTION DURING THE IMPLIED SEEK OF THE WRITE COMMAND. THE SAME OPERATION, INCLUDING THE EXPLICIT SEEK IS REPEATED FOR READ HEADER AND DATA.

FORMAT WITH MIDTRANSFER SEEK

THIS TEST WRITES MULTIPLE SECTORS STARTING WITH CYLINDER 0, TRACK 4, SECTOR 31, CAUSING A MIDTRANSFER SEEK AFTER THE FIRST SECTOR IS WRITTEN. THE SAME SECTORS ARE READ WITH READ HEADER AND DATA COMMAND.

FORMAT EACH SECTOR ADDRESS

HEADERS AND DATA OF EACH SECTOR ON CYLINDER 0, TRACK 0 ARE FORMATTED AND READ WITH THE PROGRAM VERIFYING HEADERS AND DATA.

FORMAT EACH TRACK ADDRESS

THIS TEST FORMATS SECTOR 0 OF EACH TRACK ON CYLINDER 0 AND READS EACH SECTOR WITH THE PROGRAM VERIFYING HEADERS AND DATA.

FORMAT PRIME CYLINDERS

THIS TEST FORMATS AND READS SECTOR 0, TRACK 0 ON EACH PRIME CYLINDER, I.E., CYLINDERS 1,2,4,8,...,512.

FORMAT LAST SECTOR

THIS TEST READS HEADER AND DATA ON THE LAST SECTOR, I.E., CYLINDER 822, TRACK 4, SECTOR 31, AND VERIFIES THAT LBT STATUS SETS.

FOR W/ AOE ERROR

THIS TEST READS MULTIPLE SECTORS STARTING WITH THE LAST SECTOR AND VERIFIES THAT AOE STATUS SETS.

FORMAT INVALID SECTOR ADDRESS

THIS TEST USES AN ILLEGAL SECTOR ADDRESS AND VERIFIES THAT IAE STATUS SETS.

FORMAT INVALID TRACK ADDRESS

THIS TEST USES AN ILLEGAL TRACK ADDRESS AND VERIFIES THAT IAE STATUS SETS.

FORMAT INVALID CYLINDER ADDRESS

THIS TEST USES AN ILLEGAL CYLINDER ADDRESS AND VERIFIES THAT IAE STATUS SETS.

FORMAT AT OFFSET

THE PROGRAM SETS OFFSET MODE AND EXECUTES A WRITE HEADER AND DATA COMMAND, VERIFYING THAT OFFSET MODE IS RESET BY THE WRITE COMMAND.

IVC FORMAT TEST

VOLUME VALID IS RESET BY SETTING AND RESETTING DIAGNOSTIC MODE. THE TEST THEN EXECUTES A WRITE HEADER AND DATA COMMAND AND VERIFIES THAT INVALID COMMAND STATUS SETS. THE TEST IS REPEATED FOR READ HEADER AND DATA COMMAND.

FORMAT ERROR TEST(18 AND 16)

A SINGLE SECTOR IS FORMATTED WITH THE OFFSET REGISTER SET FOR 16 BIT FORMAT AFTER WHICH THE SAME SECTOR IS READ IN 18 BIT FORMAT WITH THE PROGRAM VERIFYING THAT FORMAT ERROR STATUS IS SET. THE SAME PROCEDURE IS REPEATED WITH THE SECTOR WRITTEN IN 18 BIT FORMAT AND READ IN 16 BIT FORMAT.

FORMAT HCE (FIRST AND SECOND HEADER WORDS)

THESE TWO TESTS WRITE AN INCORRECT HEADER THEN READ THE HEADER AND VERIFY THAT THE CORRECT HEADER ERROR IS DETECTED. THE TESTS SETUP THE CORRECT HEADER, THEN COMPLEMENT BIT 0 AND USE THE MODIFIED BUFFER TO WRITE THE HEADER. THE PROCESS IS REPEATED UNTIL EACH BIT POSITION HAS BEEN SEPARATELY TESTED.

```
798                                     \
```

```
                                ;PROGRAM REVISION #001

                                .TITLE  CZRMDC0 RM03/2 FCTNL TST 2
                                ;*COPYRIGHT (C) 1977
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY DOUG RIIKONEN
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*
000001                          $TN=1
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        12            ENABLE EXTENDED STATUS
                                ;*        11            INHIBIT ITERATIONS
                                ;*        10            BELL ON ERROR
                                ;*         9            LOOP ON ERROR
                                ;*         8            LOOP ON TEST IN SWR<7:0>
                                ;*         7            TN128
                                ;*         6            TN64
                                ;*         5            TN32
                                ;*         4            TN16
                                ;*         3            TN8
                                ;*         2            TN4
                                ;*         1            TN2
                                ;*         0            TN1
                                .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100
                                .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12              ;;CODE FOR LINE FEED
000015                          CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
```

```
000003                  R3=     %3              ;;GENERAL REGISTER
000004                  R4=     %4              ;;GENERAL REGISTER
000005                  R5=     %5              ;;GENERAL REGISTER
000006                  R6=     %6              ;;GENERAL REGISTER
000007                  R7=     %7              ;;GENERAL REGISTER
000006                  SP=     %6              ;;STACK POINTER
000007                  PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
```

```
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR

                        .SBTTL  RM03 REGISTER BIT DEFINITIONS

                        ;RMCS1  CONTROL STATUS REGISTER

004000          DVA     =       BIT11           ;DEVICE AVAILABLE-READ ONLY
000040          F4      =       BIT05           ;FUNCTION CODE
000020          F3      =       BIT04           ;FUNCTION CODE
000010          F2      =       BIT03           ;FUNCTION CODE
000004          F1      =       BIT02           ;FUNCTION CODE
000002          F0      =       BIT01           ;FUNCTION CODE
000001          GO      =       BIT00           ;GO BIT
000077          FNCMSK  =       000077          ;FUNCTION CODE MASK

                        ;FUNCTION CODES (BITS 01-05 OF RMCS1)

000000          NOP     =       000000          ;NOP COMMAND
000002          ILF02   =       000002          ;ILLEGAL COMMAND
000004          SEEK    =       000004          ;SEEK COMMAND
000006          RECAL   =       000006          ;RECALIBRATE COMMAND
000010          DRVCLR  =       000010          ;DRIVE CLEAR COMMAND
000012          RLEASE  =       000012          ;RELEASE COMMAND
000014          OFFSET  =       000014          ;OFFSET COMMAND
000016          RTC     =       000016          ;RETURN TO CENTERLINE COMMAND
000020          RIP     =       000020          ;READ IN PRESET COMMAND
```

```
000022          PAKACK  =       000022          ;PACK ACKNOWLEDGE COMMAND
000022          PACACK  =       PAKACK
000024          ILF24   =       000024          ;ILLEGAL COMMAND
000026          ILF26   =       000026          ;ILLEGAL COMMAND
000030          SEARCH  =       000030          ;SEARCH COMMAND
000030          ILF30   =       000030          ;ILLEGAL COMMAND
000032          ILF32   =       000032          ;ILLEGAL COMMAND
000034          ILF34   =       000034          ;ILLEGAL COMMAND
000036          ILF36   =       000036          ;ILLEGAL COMMAND
000040          ILF40   =       000040          ;ILLEGAL COMMAND
000042          ILF42   =       000042          ;ILLEGAL COMMAND
000044          ILF44   =       000044          ;ILLEGAL COMMAND
000046          ILF46   =       000046          ;ILLEGAL COMMAND
000050          WCD     =       000050          ;WRITE CHECK DATA COMMAND
000052          WCH     =       000052          ;WRITE CHECK HEADER AND DATA
000054          ILF54   =       000054          ;ILLEGAL COMMAND
000056          ILF56   =       000056          ;ILLEGAL COMMAND
000060          WD      =       000060          ;WRITE DATA COMMAND
000062          WH      =       000062          ;WRITE HEADER AND DATA COMMAND
000064          ILF64   =       000064          ;ILLEGAL COMMAND
000066          ILF66   =       000066          ;ILLEGAL COMMAND
000070          RD      =       000070          ;READ DATA COMMAND
000072          RH      =       000072          ;READ HEADER AND DATA COMMAND
000074          ILF74   =       000074          ;ILLEGAL COMMAND
000076          ILF76   =       000076          ;ILLEGAL COMMAND

                ;RMDA   DISK ADDRESS REGISTER

002000          TA4     =       BIT10           ;TRACK ADDRESS 4
001000          TA2     =       BIT09           ;TRACK ADDRESS 2
000400          TA1     =       BIT08           ;TRACK ADDRESS 1
000020          SA16    =       BIT04           ;SECTOR ADDRESS 16
000010          SA8     =       BIT03           ;SECTOR ADDRESS 8
000004          SA4     =       BIT02           ;SECTOR ADDRESS 4
000002          SA2     =       BIT01           ;SECTOR ADDRESS 2
000001          SA1     =       BIT00           ;SECTOR ADDRESS 1

                ;TRACK,SECTOR MASKS

003400          TADMSK  =       003400          ;TRACK ADDRESS MASK
000037          SADMSK  =       000037          ;SECTOR ADDRESS MASK

                ;RMDS   DRIVE STATUS REGISTER

100000          ATA     =       BIT15           ;ATTENTION ACTIVE
040000          ERR     =       BIT14           ;COMPOSITE ERROR
020000          PIP     =       BIT13           ;POSITIONING IN PROGRESS
010000          MOL     =       BIT12           ;MEDIUM ON LINE
004000          WRL     =       BIT11           ;WRITE LOCK
002000          LBT     =       BIT10           ;LAST BLOCK TRANSFERRED
001000          PGM     =       BIT09           ;PROGRAMMABLE
000400          DPR     =       BIT08           ;DRIVE PRESENT
000200          DRY     =       BIT07           ;DRIVE READY
000100          VV      =       BIT06           ;VOLUME VALID
000001          OM      =       BIT00           ;OFFSET MODE ACTIVE
```

```
1023                            ;RMER1  ERROR REGISTER #1
1024
1025            100000          DCK     =       BIT15           ;DATA CHECK ERROR
1026            040000          UNS     =       BIT14           ;DRIVE UNSAFE
1027            020000          OPI     =       BIT13           ;OPERATION INCOMPLETE
1028            010000          DTE     =       BIT12           ;DRIVE TIMING ERROR
1029            004000          WLE     =       BIT11           ;WRITE LOCK ERROR
1030            002000          IAE     =       BIT10           ;INVALID ADDRESS ERROR
1031            001000          AOE     =       BIT09           ;ADDRESS OVERFLOW ERROR
1032            000400          HCRC    =       BIT08           ;HEADER CRC ERROR
1033            000200          HCE     =       BIT07           ;HEADER COMPARE ERROR
1034            000100          ECH     =       BIT06           ;ECC "HARD" ERROR
1035            000040          WCF     =       BIT05           ;WRITE CLOCK FAILURE
1036            000020          FER     =       BIT04           ;FORMAT ERROR
1037            000010          PAR     =       BIT03           ;PARITY ERROR
1038            000004          RMR     =       BIT02           ;REGISTER MODIFICATION REFUSED
1039            000002          ILR     =       BIT01           ;ILLEGAL REGISTER
1040            000001          ILF     =       BIT00           ;ILLEGAL FUNCTION
1041
1042            115760          NDTMSK  =       DCK!DTE!WLE!AOE!HCRC!HCE!ECH!WCF!FER
1043                            ;"NDTMSK" IS USED TO MASK ERROR REGISTER 1 DURING NON - DATA
1044                            ;COMMANDS, I.E., HOUSEKEEPING AND POSITIONING COMMANDS
1045
1046                            ;RMAS   ATTENTION SUMMARY REGISTER
1047
1048            000377          ATNMSK  =       377             ;MASK FOR ATTENTION BITS
1049
1050                            ;RMLA   LOOK AHEAD REGISTER
1051
1052            002000          SC4     =       BIT10           ;SECTOR COUNT = 16
1053            001000          SC3     =       BIT09           ;SECTOR COUNT = 8
1054            000400          SC2     =       BIT08           ;SECTOR COUNT = 4
1055            000200          SC1     =       BIT07           ;SECTOR COUNT = 2
1056            000100          SC0     =       BIT06           ;SECTOR COUNT = 1
1057
1058            003700          SCTMSK  =       003700          ;SECTOR COUNT MASK
1059
1060                            ;RMMR   MAINTENANCE REGISTER
1061
1062                            ;       WRITE ONLY BITS
1063
1064            100000          DBCK    =       BIT15           ;DEBUG CLOCK
1065            040000          DBEN    =       BIT14           ;DEBUG CLOCK ENABLE
1066            020000          DEBL    =       BIT13           ;DIAGNOSTIC END OF BLOCK
1067            010000          DTO     =       BIT12           ;DIAGNOSTIC TIMEOUT
1068            004000          MCLK    =       BIT11           ;MAINTENANCE CLOCK
1069            002000          MRD     =       BIT10           ;READ DATA
1070            001000          MUR     =       BIT09           ;UNIT READY
1071            000400          MOC     =       BIT08           ;ON CYLINDER
1072            000200          MSER    =       BIT07           ;SEEK ERROR
1073            000100          MDF     =       BIT06           ;DRIVE FAULT
1074            000040          MS      =       BIT05           ;SECTOR PULSE
1075            000010          MWP     =       BIT03           ;WRITE PROTECT
1076            000004          MI      =       BIT02           ;INDEX PULSE
1077            000002          MSC     =       BIT01           ;SECTOR COMPARE
1078            000001          DMD     =       BIT00           ;DIAGNOSTIC MODE
```

```
1079
1080                            ;       READ ONLY BITS
1081
1082    100000                  OCC     =       BIT15           ;OCCUPIED
1083    040000                  RG      =       BIT14           ;RUN AND GO
1084    020000                  EBL     =       BIT13           ;END OF BLOCK
1085    010000                  REX     =       BIT12           ;EXCEPTION
1086    004000                  ESRC    =       BIT11           ;ENABLE SEARCH
1087    002000                  PLFS    =       BIT10           ;LOOKING FOR SYNC
1088    001000                  ECRC    =       BIT09           ;ENABLE CRC OUT
1089    000400                  PDA     =       BIT08           ;DATA AREA
1090    000200                  PHA     =       BIT07           ;HEADER AREA
1091    000100                  CONT    =       BIT06           ;CONTINUE
1092    000040                  WC      =       BIT05           ;WORD CLOCK
1093    000020                  EECC    =       BIT04           ;ENABLE ECC OUT
1094    000010                  MWD     =       BIT03           ;WRITE DATA BIT
1095    000004                  LS      =       BIT02           ;LAST SECTOR
1096    000002                  LST     =       BIT01           ;LAST SECTOR AND TRACK
1097    000001                  DMD     =       BIT00           ;DIAGNOSTIC MODE
1098
1099                            ;RMDT   DRIVE TYPE REGISTER
1100
1101    100000                  NSA     =       BIT15           ;NOT SECTOR ADDRESSED=0
1102    040000                  TAP     =       BIT14           ;TAPE DRIVE = 0
1103    020000                  MOH     =       BIT13           ;MOVING HEAD = 1
1104    004000                  DRQ     =       BIT11           ;DRIVE REQUEST REQUIRED
1105
1106    020024                  SNGPRT  =       020024          ;SINGLE PORT DRIVE TYPE
1107    024024                  DULPRT  =       024024          ;DUAL PORT DRIVE TYPE
1108
1109                            ;RMOF   OFFSET REGISTER
1110
1111    010000                  FMT16   =       BIT12           ;16 BIT WORD FORMAT
1112    004000                  ECI     =       BIT11           ;ECC INHIBIT
1113    002000                  HCI     =       BIT10           ;HEADER COMPARE INHIBIT
1114    000200                  OFD     =       BIT07           ;OFFSET FORWARD
1115
1116
1117                            ;RMDC   DESIRED CYLINDER ADDRESS REGISTER
1118    001777                  CYLMSK  =       1777            ;MASK FOR CYLINDER ADDRESS
1119
1120                            ;RMMR2  MAINTENANCE REGISTER #2
1121
1122                            ;       READ ONLY BITS
1123    100000                  RQA     =       BIT15           ;PORT A REQUEST
1124    040000                  RQB     =       BIT14           ;PORT B REQUEST
1125    020000                  TAG     =       BIT13           ;TAG CONTROL
1126    010000                  TST     =       BIT12           ;COMMAND SEQUENCE TEST BIT
1127    004000                  CC      =       BIT11           ;CONTROL OR CYLINDER TAG
1128    002000                  CH      =       BIT10           ;CONTROL OR HEAD TAG
1129    001000                  BB09    =       BIT09           ;TAG BUS
1130    000400                  BB08    =       BIT08           ;TAG BUS
1131    000200                  BB07    =       BIT07           ;TAG BUS
1132    000100                  BB06    =       BIT06           ;TAG BUS
1133    000040                  BB05    =       BIT05           ;TAG BUS
1134    000020                  BB04    =       BIT04           ;TAG BUS
```

```
1135    000010          BB03    =       BIT03           ;TAG BUS
1136    000004          BB02    =       BIT02           ;TAG BUS
1137    000002          BB01    =       BIT01           ;TAG BUS
1138    000001          BB00    =       BIT00           ;TAG BUS
1139
1140
1141                    ;RMER2  ERROR REGISTER 2
1142
1143    100000          BSE     =       BIT15           ;BAD SECTOR ERROR
1144    040000          SKI     =       BIT14           ;SEEK INCOMPLETE
1145    020000          OPE     =       BIT13           ;OPERATOR PLUG ERROR
1146    010000          IVC     =       BIT12           ;INVALID COMMAND ERROR
1147    004000          LSC     =       BIT11           ;LOSS OF SYSTEM CLOCK
1148    002000          LBC     =       BIT10           ;LOSS OF BIT CLOCK
1149    000200          DVC     =       BIT07           ;DEVICE CHECK
1150    000010          DPE     =       BIT03           ;DATA PARITY ERROR
1151
1152                    .SBTTL  PROGRAM MNEMONICS
1153
1154    100000          MSE     =       BIT15           ;MANUFACTURING DETECTED SECTOR ERROR
1155    040000          USE     =       BIT14           ;USER DETECTED SECTOR ERROR
1156
1157                    .SBTTL  RM03 REGISTER INDEX VALUES
1158
1159    000000          RMCSI   =       00              ;CONTROL STATUS REGISTER
1160    000006          RMDA    =       06              ;DISK ADDRESS REGISTER
1161    000012          RMDS    =       12              ;DRIVE STATUS REGISTER
1162    000014          RMER1   =       14              ;ERROR REGISTER 1
1163    000016          RMAS    =       16              ;ATTENTION SUMMARY REGISTER
1164    000020          RMLA    =       20              ;LOOK AHEAD REGISTER
1165    000024          RMMR1   =       24              ;MAINTENANCE REGISTER
1166    000026          RMDT    =       26              ;DRIVE TYPE REGISTER
1167    000030          RMSN    =       30              ;SERIAL NUMBER REGISTER
1168    000032          RMOF    =       32              ;OFFSET REGISTER
1169    000034          RMDC    =       34              ;DESIRED CYLINDER REGISTER
1170    000036          RMCC    =       36              ;CURRENT CYLINDER REGISTER
1171    000040          RMMR2   =       40              ;MAINTENANCE REGISTER 2
1172    000042          RMER2   =       42              ;ERROR REGISTER 2
1173    000044          RMEC1   =       44              ;ECC POSITION REGISTER
1174    000046          RMEC2   =       46              ;ECC PATTERN REGISTER
1175    000050          ILRG50  =       50              ;ILLEGAL REGISTER 50
1176    000052          ILRG52  =       52              ;ILLEGAL REGISTER 52
1177    000054          ILRG54  =       54              ;ILLEGAL REGISTER 54
1178    000056          ILRG56  =       56              ;ILLEGAL REGISTER 56
1179    000060          ILRG60  =       60              ;ILLEGAL REGISTER 60
1180    000062          ILRG62  =       62              ;ILLEGAL REGISTER 62
1181    000064          ILRG64  =       64              ;ILLEGAL REGISTER 64
1182    000066          ILRG66  =       66              ;ILLEGAL REGISTER 66
1183    000070          ILRG70  =       70              ;ILLEGAL REGISTER 70
1184    000072          ILRG72  =       72              ;ILLEGAL REGISTER 72
1185    000074          ILRG74  =       74              ;ILLEGAL REGISTER 74
1186    000076          ILRG76  =       76              ;ILLEGAL REGISTER 76
1187
1188
1189    000077          IDXMSK  =       77              ;MASK FOR REGISTER INDEX NUMBER
1190
```

```
1191                            .SBTTL  RH CONTROLLER REGISTER BIT DEFINITIONS
1192
1193                            ;RMCS1  CONTROL STATUS REGISTER #1
1194
1195            100000          SC      =       BIT15           ;SPECIAL CONDITION-READ ONLY
1196            040000          TRE     =       BIT14           ;TRANSFER ERROR
1197            020000          MCPE    =       BIT13           ;MASSBUS CONTROL BUS PARITY
1198                                                            ;ERROR-READ ONLY
1199            002000          PSEL    =       BIT10           ;PORT B SELECT
1200            001000          A17     =       BIT09           ;ADDRESS EXTENSION
1201            000400          A16     =       BIT08           ;ADDRESS EXTENSION
1202            000200          RDY     =       BIT07           ;READY-READ ONLY
1203            000100          IE      =       BIT06           ;INTERRUPT ENABLE
1204
1205                            ;RMCS2  RH CONTROL STATUS REGISTER #2
1206
1207            100000          DLT     =       BIT15           ;DATA LATE-READ ONLY
1208            040000          WCE     =       BIT14           ;WRITE CHECK ERROR-READ ONLY
1209            020000          UPE     =       BIT13           ;UNIBUS PARITY ERROR
1210            010000          NED     =       BIT12           ;NONEXISTANT DRIVE-READ ONLY
1211            004000          NEM     =       BIT11           ;NONEXISTANT MEMORY-READ ONLY
1212            002000          PGE     =       BIT10           ;PROGRAM ERROR-READ ONLY
1213            001000          MXF     =       BIT09           ;MISSED TRANSFER
1214            000400          MDPE    =       BIT08           ;MASSBUS DATA BUS PARITY
1215                                                            ;ERROR-READ ONLY
1216            000200          OR      =       BIT07           ;OUTPUT READY-READ ONLY
1217            000100          IR      =       BIT06           ;INPUT READY-READ ONLY
1218            000040          CLR     =       BIT05           ;CONTROLLER CLEAR
1219            000020          PAT     =       BIT04           ;PARITY TEST
1220            000010          BAI     =       BIT03           ;UNIBUS ADDRESS INCREMENT
1221                                                            ;INHIBIT
1222            000004          U2      =       BIT02           ;UNIT SELECT
1223            000002          U1      =       BIT01           ;UNIT SELECT
1224            000001          U0      =       BIT00           ;UNIT SELECT
1225
1226                            ;UNIT SELECT MASK
1227            000007          UNTMSK  =       7               ;UNIT SELECT MASK
1228
1229                            ;RMCS3  RH70 CONTROL STATUS REGISTER #3
1230            100000          APE     =       BIT15           ;ADDRESS PARITY ERROR
1231            040000          DPEHI   =       BIT14           ;DATA PARITY ERROR HIGH WORD
1232            020000          DPELO   =       BIT13           ;DATA PARITY ERROR LOW WORD
1233            010000          WCEHI   =       BIT12           ;WRITE CHECK ERROR HIGH WORD
1234            004000          WCELO   =       BIT11           ;WRITE CHECK ERROR LOW WORD
1235            002000          DBL     =       BIT10           ;DOUBLE WORD TRANSFER
1236            000100          IE      =       BIT06           ;INTERRUPT ENABLE
1237            000010          IPCK3   =       BIT03           ;INVERT PARITY CHECK
1238            000004          IPCK2   =       BIT02           ;INVERT PARITY CHECK
1239            000002          IPCK1   =       BIT01           ;INVERT PARITY CHECK
1240            000001          IPCK0   =       BIT00           ;INVERT PARITY CHECK
1241                            .SBTTL  RH CONTROLLER REGISTER INDEX VALUES
1242
1243            000000          RMCS1   =       00              ;CONTROL, STATUS REGISTER
1244            000002          RMWC    =       02              ;WORD COUNT REGISTER
1245            000004          RMBA    =       04              ;BUS ADDRESS REGISTER
1246            000010          RMCS2   =       10              ;CONTROLLER STATUS REGISTER
```

```
1247            000022                  RMDB    =       22              ;DATA BUFFER
1248            000050                  RMBAE   =       50              ;BUS ADDRESS EXTENSION
1249            000052                  RMCS3   =       52              ;CONTROL STATUS REGISTER #3
1250
1251            176700                  ABASE   =       176700          ;UNIBUS ADDRESS
1252            120254                  AVECT1  =       120254          ;UNIBUS VECTOR ADDRESS AND PRIORITY
1253
1254
1255                                    .SBTTL  TRAP CATCHER
1256
1257            000000                          .=0
1258                                    ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1259                                    ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1260                                    ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1261            000174                          .=174
1262    000174  000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
1263    000176  000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
1264
1265
1266                                    .SBTTL  ACT11 HOOKS
1267
1268                                    ;;**************************************************************
1269                                    ;HOOKS REQUIRED BY ACT11
1270            000200                          $SVPC=.                 ;SAVE PC
1271            000046                          .=46
1272    000046  033156                  $ENDAD                          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
1273            000052                          .=52
1274    000052  000000                  .WORD   0                       ;;2)SET LOC.52 TO ZERO
1275            000200                          .=$SVPC                 ;; RESTORE PC
1276
1277                                    .SBTTL  STARTING ADDRESS
1278
1279                                    ;THE PROGRAM STARTS AT LOCATION 200
1280            000200                  .       =       200
1281    000200  000137  005324                  JMP     START           ;JUMP TO START OF PROGRAM
1282
1283            001100                  .=1100
1284                                    .SBTTL  APT PARAMETER BLOCK
1285
1286                                    ;;**************************************************************
1287                                    ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1288                                    ;;**************************************************************
1289            001100                          .$X=.   ;;SAVE CURRENT LOCATION
1290            000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1291    000024  000200                  200     ;;FOR APT START UP
1292            000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
1293    000044  001100                  $APTHDR ;;POINT TO APT HEADER BLOCK
1294            001100                          .=.$X   ;;RESET LOCATION COUNTER
1295                                    ;;**************************************************************
1296                                    ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1297                                    ;INTERFACE SPEC.
1298
1299    001100                          $APTHD:
1300    001100  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1301    001102  001222                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1302    001104  000001                  $TSTM:  .WORD   1       ;;RUN TIM OF LONGEST TEST
```

```
 1303  001106  000002                $PASTM: .WORD   2       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
 1304  001110  000002                $UNITM: .WORD   2       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
 1305  001112  000042                        .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
 1306          001114                TAGADR  =       .
```

```
1307                                    .SBTTL  COMMON TAGS
1308
1309                                    ;;**********************************************************************
1310                                    ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
1311                                    ;*USED IN THE PROGRAM.
1312
1313            001114                          .=TAGADR
1314  001114                            $CMTAG:                         ;;START OF COMMON TAGS
1315  001114  000000                           .WORD   0
1316  001116     000                    $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
1317  001117     000                    $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
1318  001120  000000                    $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
1319  001122  000000                    $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
1320  001124  000000                    $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
1321  001126  000000                    $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
1322  001130     000                    $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
1323  001131     001                    $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
1324  001132  000000                    $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
1325  001134  000000                    $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
1326  001136  000000                    $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
1327  001140  000000                    $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
1328  001142  000000                    $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
1329  001144  000000                            .WORD   0               ;;RESERVED--NOT TO BE USED
1330  001146  000000                            .WORD   0
1331  001150     000                    $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
1332  001151     000                    $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
1333  001152  000000                            .WORD   0
1334  001154  177570                    SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
1335  001156  177570                    DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
1336  001160  177560                    $TKS:   177560                  ;;TTY KBD STATUS
1337  001162  177562                    $TKB:   177562                  ;;TTY KBD BUFFER
1338  001164  177564                    $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
1339  001166  177566                    $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
1340  001170     000                    $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
1341  001171     002                    $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
1342  001172     012                    $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
1343  001173     000                    $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
1344  001174  000000                    $TMP0:  .WORD   0               ;;USER DEFINED
1345  001176  000000                    $TMP1:  .WORD   0               ;;USER DEFINED
1346  001200  000000                    $TMP2:  .WORD   0               ;;USER DEFINED
1347  001202  000000                    $TMP3:  .WORD   0               ;;USER DEFINED
1348  001204  000000                    $TMP4:  .WORD   0               ;;USER DEFINED
1349  001206  000000                    $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
1350  001210  000000                    $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
1351  001212  177607  000377            $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
1352  001216     077                    $QUES:  .ASCII  /?/             ;;QUESTION MARK
1353  001217     015                    $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
1354  001220  000012                    $LF:    .ASCIZ  <12>            ;;LINE FEED
1355                                    ;;**********************************************************************
1356                                    .SBTTL  APT MAILBOX-ETABLE
1357
1358                                    ;;**********************************************************************
1359                                    .NLIST ME
1360                                    ;
1361                                    ;
1362
```

```
1363                                    .EVEN
1364   001222                           $MAIL:                          ;;APT MAILBOX
1365   001222  000000                   $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
1366   001224  000000                   $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
1367   001226  000000                   $TESTN: .WORD   ATESTN          ;;TEST NUMBER
1368   001230  000000                   $PASS:  .WORD   APASS           ;;PASS COUNT
1369   001232  000000                   $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
1370   001234  000000                   $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
1371   001236  000000                   $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
1372   001240  000000                   $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
1373   001242                           $ETABLE:                        ;;APT ENVIRONMENT TABLE
1374   001242     000                   $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
1375   001243     000                   $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
1376   001244  000000                   $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
1377   001246  000000                   $USWR:  .WORD   AUSWR           ;;USER SWITCHES
1378   001250  000000                   $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
1379                                    ;*                              BITS 15-11=CPU TYPE
1380                                    ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1381                                    ;*                                      11/70=06,PDQ=07,Q=10
1382                                    ;*                              BIT 10=REAL TIME CLOCK
1383                                    ;*                              BIT  9=FLOATING POINT PROCESSOR
1384                                    ;*                              BIT  8=MEMORY MANAGEMENT
1385   001252     000                   $MAMS1: .BYTE   AMAMS1          ;;HIGH ADDRESS,M.S. BYTE
1386   001253     000                   $MTYP1: .BYTE   AMTYP1          ;;MEM. TYPE,BLK#1
1387                                    ;*                              MEM.TYPE BYTE   --  (HIGH BYTE)
1388                                    ;*                                      900 NSEC CORE=001
1389                                    ;*                                      300 NSEC BIPOLAR=002
1390                                    ;*                                      500 NSEC MOS=003
1391   001254  000000                   $MADR1: .WORD   AMADR1           ;;HIGH ADDRESS,BLK#1
1392                                    ;*                                MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
1393   001256     000                   $MAMS2: .BYTE   AMAMS2          ;;HIGH ADDRESS,M.S. BYTE
1394   001257     000                   $MTYP2: .BYTE   AMTYP2          ;;MEM.TYPE,BLK#2
1395   001260  000000                   $MADR2: .WORD   AMADR2          ;;MEM.LAST ADDRESS,BLK#2
1396   001262     000                   $MAMS3: .BYTE   AMAMS3          ;;HIGH ADDRESS,M.S.BYTE
1397   001263     000                   $MTYP3: .BYTE   AMTYP3          ;;MEM.TYPE,BLK#3
1398   001264  000000                   $MADR3: .WORD   AMADR3          ;;MEM.LAST ADDRESS,BLK#3
1399   001266     000                   $MAMS4: .BYTE   AMAMS4          ;;HIGH ADDRESS,M.S.BYTE
1400   001267     000                   $MTYP4: .BYTE   AMTYP4          ;;MEM.TYPE,BLK#4
1401   001270  000000                   $MADR4: .WORD   AMADR4          ;;MEM.LAST ADDRESS,BLK#4
1402   001272  120254                   $VECT1: .WORD   AVECT1          ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1403   001274  000000                   $VECT2: .WORD   AVECT2          ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1404   001276  176700                   $BASE:  .WORD   ABASE           ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
1405   001300  000000                   $DEVM:  .WORD   ADEVM           ;;DEVICE MAP
1406   001302  000000                   $CDW1:  .WORD   ACDW1           ;;CONTROLLER DESCRIPTION WORD#1
1407   001304  000000                   $CDW2:  .WORD   ACDW2           ;;CONTROLLER DESCRIPTION WORD#2
1408   001306  000000                   $DDW0:  .WORD   ADDW0           ;;DEVICE DESCRIPTOR WORD#0
1409   001310  000000                   $DDW1:  .WORD   ADDW1           ;;DEVICE DESCRIPTOR WORD#1
1410   001312  000000                   $DDW2:  .WORD   ADDW2           ;;DEVICE DESCRIPTOR WORD#2
1411   001314  000000                   $DDW3:  .WORD   ADDW3           ;;DEVICE DESCRIPTOR WORD#3
1412   001316  000000                   $DDW4:  .WORD   ADDW4           ;;DEVICE DESCRIPTOR WORD#4
1413   001320  000000                   $DDW5:  .WORD   ADDW5           ;;DEVICE DESCRIPTOR WORD#5
1414   001322  000000                   $DDW6:  .WORD   ADDW6           ;;DEVICE DESCRIPTOR WORD#6
1415   001324  000000                   $DDW7:  .WORD   ADDW7           ;;DEVICE DESCRIPTOR WORD#7
1416   001326                           $ETEND:
1417                                    .MEXIT
1418   001326  000000                   CTLFG:  .WORD   0                ;CONTROL-C FLAG
```

```
1419
1420                                    ;THE REGISTER INPUT BUFFER IS USED FOR
1421                                    ;STORING DRIVE STATUS
1422
1423  001330                            GETBUF:
1424
1425                                    ;REGISTER INPUT BUFFER
1426  001330  000000                    RMCS1I: .WORD                   ;CONTROL,STATUS REGISTER
1427  001332  000000                    RMWCI: .WORD                    ;WORD COUNT REGISTER
1428  001334  000000                    RMBAI: .WORD                    ;BUS ADDRESS REGISTER
1429  001336  000000                    RMDAI: .WORD                    ;DISK ADDRESS REGISTER
1430  001340  000000                    RMCS2I: .WORD                   ;CONTROLLER STATUS REGISTER
1431  001342  000000                    RMDSI: .WORD                    ;DRIVE STATUS REGISTER
1432  001344  000000                    RMER1I: .WORD                   ;ERROR REGISTER 1
1433  001346  000000                    RMASI: .WORD                    ;ATTENTION SUMMARY REGISTER
1434  001350  000000                    RMLAI: .WORD                    ;LOOK AHEAD REGISTER
1435  001352  000000                    RMDBI: .WORD                    ;DATA BUFFER
1436  001354  000000                    RMMR1I: .WORD                   ;MAINTENANCE REGISTER #1
1437  001356  000000                    RMDTI: .WORD                    ;DRIVE TYPE REGISTER
1438  001360  000000                    RMSNI: .WORD                    ;SERIAL NUMBER REGISTER
1439  001362  000000                    RMOFI: .WORD                    ;OFFSET REGISTER
1440  001364  000000                    RMDCI: .WORD                    ;DESIRED CYLINDER REGISTER
1441  001366  000000                    RMCCI: .WORD                    ;CURRENT CYLINDER REGISTER
1442  001370  000000                    RMMR2I: .WORD                   ;MAINTENANCE REGISTER #2
1443  001372  000000                    RMER2I: .WORD                   ;ERROR REGISTER 2
1444  001374  000000                    RMEC1I: .WORD                   ;ECC POSITION REGISTER
1445  001376  000000                    RMEC2I: .WORD                   ;ECC PATTERN REGISTER
1446
1447                                    ;THE REGISTER OUTPUT BUFFER IS USED FOR
1448                                    ;ASSEMBLING DATA GOING TO REGISTER
1449
1450  001400                            PUTBUF:
1451
1452                                    ;REGISTER OUTPUT BUFFER
1453  001400  000000                    RMCS1O: .WORD                   ;CONTROL,STATUS REGISTER
1454  001402  000000                    RMWCO: .WORD                    ;WORD COUNT REGISTER
1455  001404  000000                    RMBAO: .WORD                    ;BUS ADDRESS REGISTER
1456  001406  000000                    RMDAO: .WORD                    ;DISK ADDRESS REGISTER
1457  001410  000000                    RMCS2O: .WORD                   ;CONTROLLER STATUS REGISTER
1458  001412  000000                    RMDSO: .WORD                    ;DRIVE STATUS REGISTER
1459  001414  000000                    RMER1O: .WORD                   ;ERROR REGISTER 1
1460  001416  000000                    RMASO: .WORD                    ;ATTENTION SUMMARY REGISTER
1461  001420  000000                    RMLAO: .WORD                    ;LOOK AHEAD REGISTER
1462  001422  000000                    RMDBO: .WORD                    ;DATA BUFFER
1463  001424  000000                    RMMR1O: .WORD                   ;MAINTENANCE REGISTER #1
1464  001426  000000                    RMDTO: .WORD                    ;DRIVE TYPE REGISTER
1465  001430  000000                    RMSNO: .WORD                    ;SERIAL NUMBER REGISTER
1466  001432  000000                    RMOFO: .WORD                    ;OFFSET REGISTER
1467  001434  000000                    RMDCO: .WORD                    ;DESIRED CYLINDER REGISTER
1468  001436  000000                    RMCCO: .WORD                    ;CURRENT CYLINDER REGISTER
1469  001440  000000                    RMMR2O: .WORD                   ;MAINTENANCE REGISTER #2
1470  001442  000000                    RMER2O: .WORD                   ;ERROR REGISTER 2
1471  001444  000000                    RMEC1O: .WORD                   ;ECC POSITION REGISTER
1472  001446  000000                    RMEC2O: .WORD                   ;ECC PATTERN REGISTER
1473
1474                                    ;EACH WORD OF THE TEST QUE CONTAINS THE DEVICE NUMBER IN
```

```
1475                                ;THE LOW BYTE AND THE ATTENTION BIT IN THE HIGH BYTE. A ZERO
1476                                ;WORD IS A BLANK AND REPRESENTS THE END OF THE QUE.
1477
1478  001450  000012                TSTQUE: .BLKW   10.      ;TEST QUE
1479
1480                                ;MEDIA ENABLE IS SET IF THE BAD SECTOR FILES HAVE BEEN RECOVERED
1481                                ;FOR THE UNIT UNDER TEST, OTHERWISE IT IS ZERO.
1482  001474  000000                MEDENB: .WORD           ;MEDIA ENABLE
1483
1484                                ;LOCATIONS "ASNDC" AND "ASNDC" CONTAIN THE CYLINDER, TRACK AND SECTOR
1485                                ;ADDRESS ASSIGNED BY THE BAD SECTOR MODULE.
1486  001476  000000                ASNDC:  .WORD           ;ASSIGNED DESIRED CYLINDER
1487  001500  000000                ASNDA:  .WORD           ;ASSIGNED TRACK, AND SECTOR
1488  001502  000000                CLKADR: .WORD           ;UNIBUS ADDRESS OF KW11 CLOCK
1489  001504  000000                CLKVCT: .WORD           ;VECTOR ADDRESS OF KW11 CLOCK
1490
1491                                ;THE GET INDEX TABLE CONTAINS A BYTE LIST OF REGISTERS WHICH
1492                                ;ARE READ BY THE GET SUBROUTINE.  THE LIST IS TERMINATED BY
1493                                ;A NEGATIVE BYTE.
1494  001506  000027                GETINX: .BLKB   23.             ;GET INDEX TABLE
1495
1496                                ;THE PUT INDEX TABLE ICONTAINS A BYTE LIST OF REGISTERS WHICH
1497                                ;ARE WRITTEN BY THE PUT SUBROUTINE.  THE LIST IS TERMINATED BY
1498                                ;A NEGATIVE BYTE.
1499  001535  000027                PUTINX: .BLKB   23.             ;PUT INDEX TABLE
1500
1501                                ;PUT TAGS HERE
1502
```

```
1503                                    .SBTTL  ERROR POINTER TABLE
1504
1505                                    ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
1506                                    ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
1507                                    ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
1508                                    ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
1509                                    ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
1510
1511                                    ;*      EM              ;;POINTS TO THE ERROR MESSAGE
1512                                    ;*      DH              ;;POINTS TO THE DATA HEADER
1513                                    ;*      DT              ;;POINTS TO THE DATA
1514                                    ;*      DF              ;;POINTS TO THE DATA FORMAT
1515
1516
1517  001564                            $ERRTB:
1518
1519
```

```
1520
1521  001564  065430                ;ERROR  1       WRONG UNIT SELECTED
1522  001566  071522                        EMT1
1523  001570  071646                        EHT1
1524  001572  071736                        EDT1
1525                                  EFT1
1526
1527                                ;ERROR  2       DEVICE WENT UNAVAILABLE
1528  001574  065434                        EMT2
1529  001576  071522                        EHT1
1530  001600  071646                        EDT1
1531  001602  071736                        EFT1
1532
1533
1534                                ;ERROR  3       DEVICE WENT NONEXISTENT
1535  001604  065442                        EMT3
1536  001606  071522                        EHT1
1537  001610  071646                        EDT1
1538  001612  071736                        EFT1
1539
1540
1541                                ;ERROR  4       CONTROLLER NOT READY
1542  001614  065450                        EMT4
1543  001616  071522                        EHT1
1544  001620  071646                        EDT1
1545  001622  071736                        EFT1
1546
1547
1548                                ;ERROR  5       DRIVE NOT READY AND GO NOT RESET
1549  001624  065456                        EMT5
1550  001626  071522                        EHT1
1551  001630  071646                        EDT1
1552  001632  071736                        EFT1
1553
1554
1555                                ;ERROR  6       UNEXPECTED VALUE FOR "ATA" STATUS
1556  001634  065464                        EMT6
1557  001636  071522                        EHT1
1558  001640  071646                        EDT1
1559  001642  071736                        EFT1
1560
1561
1562                                ;ERROR  7       BUS TIMEOUT TRYING TO READ OR WRITE REGISTER
1563  001644  065472                        EMT7
1564  001646  000000                        0
1565  001650  000000                        0
1566  001652  000000                        0
1567
1568
1569                                ;ERROR  10      DRIVE NOT READY BUT GO IS RESET
1570  001654  065500                        EMT10
1571  001656  071522                        EHT1
1572  001660  071646                        EDT1
1573  001662  071736                        EFT1
1574
1575
```

```
1576
1577  001664  065504                ;ERROR  11      GO NOT RESET BUT DRIVE IS READY
1578  001666  071522                        EMT11
1579  001670  071646                        EHT1
1580  001672  071736                        EDT1
1581                                  EFT1
1582
1583                                ;ERROR  12      INCORRECT FUNCTION CODE
1584  001674  065510                        EMT12
1585  001676  071522                        EHT1
1586  001700  071646                        EDT1
1587  001702  071736                        EFT1
1588
1589
1590                                ;ERROR  13      PARITY ERROR READING REMOTE REGISTERS
1591  001704  065516                        EMT13
1592  001706  071522                        EHT1
1593  001710  071646                        EDT1
1594  001712  071736                        EFT1
1595
1596
1597                                ;ERROR  14      TRANSFER ERROR IS INCORRECT
1598  001714  065530                        EMT14
1599  001716  071522                        EHT1
1600  001720  071646                        EDT1
1601  001722  071736                        EFT1
1602
1603
1604                                ;ERROR  15      INCORRECT WORD COUNT
1605  001724  065536                        EMT15
1606  001726  071522                        EHT1
1607  001730  071646                        EDT1
1608  001732  071736                        EFT1
1609
1610
1611                                ;ERROR  16      INCORRECT BUS ADDRESS
1612  001734  065544                        EMT16
1613  001736  071522                        EHT1
1614  001740  071646                        EDT1
1615  001742  071736                        EFT1
1616
1617
1618                                ;ERROR  17      INCORRECT LBT STATUS
1619  001744  065554                        EMT17
1620  001746  071522                        EHT1
1621  001750  071646                        EDT1
1622  001752  071736                        EFT1
1623
1624
1625                                ;ERROR  20      INCORRECT AOE
1626  001754  065564                        EMT20
1627  001756  071522                        EHT1
1628  001760  071646                        EDT1
1629  001762  071736                        EFT1
1630
1631
```

```
1632
1633  001764  065574                ;ERROR  21      INCORRECT DISK ADDRESS
1634  001766  071522                        EMT21
1635  001770  071646                        EHT1
1636  001772  071736                        EDT1
1637                                  EFT1
1638
1639                                ;ERROR  22      INCORRECT CYLINDER ADDRESS
1640  001774  065604                        EMT22
1641  001776  071522                        EHT1
1642  002000  071646                        EDT1
1643  002002  071736                        EFT1
1644
1645
1646                                ;ERROR  23      INCORRECT WLE STATUS
1647  002004  065614                        EMT23
1648  002006  071522                        EHT1
1649  002010  071646                        EDT1
1650  002012  071736                        EFT1
1651
1652
1653                                ;ERROR  24      INCORRECT UPE STATUS
1654  002014  065624                        EMT24
1655  002016  071522                        EHT1
1656  002020  071646                        EDT1
1657  002022  071736                        EFT1
1658
1659
1660                                ;ERROR  25      INCORRECT WCF STATUS
1661  002024  065634                        EMT25
1662  002026  071522                        EHT1
1663  002030  071646                        EDT1
1664  002032  071736                        EFT1
1665
1666
1667                                ;ERROR  26      INCORRECT WCE STATUS
1668  002034  065644                        EMT26
1669  002036  071522                        EHT1
1670  002040  071646                        EDT1
1671  002042  071736                        EFT1
1672
1673
1674                                ;ERROR  27      INCORRECT MDPE STATUS
1675  002044  065654                        EMT27
1676  002046  071522                        EHT1
1677  002050  071646                        EDT1
1678  002052  071736                        EFT1
1679
1680
1681                                ;ERROR  30      INCORRECT DCK STATUS
1682  002054  065664                        EMT30
1683  002056  071522                        EHT1
1684  002060  071646                        EDT1
1685  002062  071736                        EFT1
1686
1687
```

```
1688
1689  002064  065674                ;ERROR  31      INCORRECT ECH STATUS
1690  002066  071522                        EMT31
1691  002070  071646                        EHT1
1692  002072  071736                        EDT1
1693                                  EFT1
1694
1695                                ;ERROR  32      DLT SHOULD NOT BE SET
1696  002074  065704                        EMT32
1697  002076  071522                        EHT1
1698  002100  071646                        EDT1
1699  002102  071736                        EFT1
1700
1701
1702                                ;ERROR  33      MXF SHOULD NOT BE SET
1703  002104  065714                        EMT33
1704  002106  071522                        EHT1
1705  002110  071646                        EDT1
1706  002112  071736                        EFT1
1707
1708
1709                                ;ERROR  34      DTE SHOULD NOT BE SET
1710  002114  065724                        EMT34
1711  002116  071522                        EHT1
1712  002120  071646                        EDT1
1713  002122  071736                        EFT1
1714
1715
1716                                ;ERROR  35      INCORRECT HCRC STATUS
1717  002124  065734                        EMT35
1718  002126  071522                        EHT1
1719  002130  071646                        EDT1
1720  002132  071736                        EFT1
1721
1722
1723                                ;ERROR  36      INCORRECT HCE STATUS
1724  002134  065744                        EMT36
1725  002136  071522                        EHT1
1726  002140  071646                        EDT1
1727  002142  071736                        EFT1
1728
1729
1730                                ;ERROR  37      INCORRECT FER STATUS
1731  002144  065754                        EMT37
1732  002146  071522                        EHT1
1733  002150  071646                        EDT1
1734  002152  071736                        EFT1
1735
1736
1737                                ;ERROR  40      DPE SHOULD NOT BE SET (NOT A DATA COMMAND)
1738  002154  065764                        EMT40
1739  002156  071522                        EHT1
1740  002160  071646                        EDT1
1741  002162  071736                        EFT1
1742
1743
```

```
1744
1745  002164  065772                    ;ERROR  41      LOST "MOL" DURING PACK ACKNOWLEDGE
1746  002166  071522                            EMT41
1747  002170  071646                            EHT1
1748  002172  071736                            EDT1
1749                                      EFT1
1750
1751                                      ;ERROR  42      UNSAFE ERROR DURING PACK ACKNOWLEDGE
1752  002174  066002                            EMT42
1753  002176  071522                            EHT1
1754  002200  071646                            EDT1
1755  002202  071736                            EFT1
1756
1757
1758                                      ;ERROR  43      "OPI" ERROR DURING PACK ACKNOWLEDGE
1759  002204  066014                            EMT43
1760  002206  071522                            EHT1
1761  002210  071646                            EDT1
1762  002212  071736                            EFT1
1763
1764
1765                                      ;ERROR  44      "RMR" ERROR DURING PACK ACKNOWLEDGE
1766  002214  066024                            EMT44
1767  002216  071522                            EHT1
1768  002220  071646                            EDT1
1769  002222  071736                            EFT1
1770
1771
1772                                      ;ERROR  45      "ILR" ERROR DURING PACK ACKNOWLEDGE
1773  002224  066034                            EMT45
1774  002226  071522                            EHT1
1775  002230  071646                            EDT1
1776  002232  071736                            EFT1
1777
1778
1779                                      ;ERROR  46      "ILF" ERROR DURING PACK ACKNOWLEDGE
1780  002234  066044                            EMT46
1781  002236  071522                            EHT1
1782  002240  071646                            EDT1
1783  002242  071736                            EFT1
1784
1785
1786                                      ;ERROR  47      COMPOSITE ERROR STATUS IS INCORRECT
1787  002244  066054                            EMT47
1788  002246  071522                            EHT1
1789  002250  071646                            EDT1
1790  002252  071736                            EFT1
1791
1792
1793                                      ;ERROR  50      PARITY ERROR WRITING REMOTE REGISTERS
1794  002254  066062                            EMT50
1795  002256  071522                            EHT1
1796  002260  071646                            EDT1
1797  002262  071736                            EFT1
1798
1799
```

```
1800                                    ;ERROR  51      INCORRECT IAE STATUS DURING SEEK COMMAND
1801  002264  066072                              EMT51
1802  002266  071522                              EHT1
1803  002270  071646                              EDT1
1804  002272  071736                              EFT1
1805
1806
1807                                    ;ERROR  52      OPI ERROR DURING SEEK - MEDIUM IS NOT ON LINE
1808  002274  066104                              EMT52
1809  002276  071522                              EHT1
1810  002300  071646                              EDT1
1811  002302  071736                              EFT1
1812
1813
1814                                    ;ERROR  53      OPI ERROR DURING SEEK - MEDIUM IS ON LINE, ASSUME
1815                                    ;               ON CYLINDER LATCH DIDN'T RESET
1816  002304  066122                              EMT53
1817  002306  071522                              EHT1
1818  002310  071646                              EDT1
1819  002312  071736                              EFT1
1820
1821
1822                                    ;ERROR  54      SEEK INCOMPLETE ERROR DURING SEEK COMMAND
1823  002314  066140                              EMT54
1824  002316  071522                              EHT1
1825  002320  071646                              EDT1
1826  002322  071736                              EFT1
1827
1828
1829                                    ;ERROR  55      DEVICE CHECK DURING SEEK COMMAND
1830  002324  066150                              EMT55
1831  002326  071522                              EHT1
1832  002330  071646                              EDT1
1833  002332  071736                              EFT1
1834
1835
1836                                    ;ERROR  56      PIP IS STILL SET AFTER SEEK - SKI IS RESET
1837  002334  066162                              EMT56
1838  002336  071522                              EHT1
1839  002340  071646                              EDT1
1840  002342  071736                              EFT1
1841
1842
1843                                    ;ERROR  57      ATA DID NOT SET DURING SEEK COMMAND
1844  002344  066200                              EMT57
1845  002346  071522                              EHT1
1846  002350  071646                              EDT1
1847  002352  071736                              EFT1
1848
1849
1850                                    ;ERROR  60      IVC ERROR DURING SEEK COMMAND - LOST
1851                                    ;               VOLUME VALID
1852  002354  066210                              EMT60
1853  002356  071522                              EHT1
1854  002360  071646                              EDT1
1855  002362  071736                              EFT1
```

```
1856
1857
1858                                    ;ERROR  61      ERRONEOUS IVC ERROR DURING SEEK COMMAND -
1859                                    ;               VOLUME VALID IS STIL SET
1860  002364  066226                                    EMT61
1861  002366  071522                                    EHT1
1862  002370  071646                                    EDT1
1863  002372  071736                                    EFT1
1864
1865
1866                                    ;ERROR  62      MOL IS ZERO, BUT OPI WAS NOT
1867                                    ;               REPORTED DURING SEEK COMMAND
1868  002374  066246                                    EMT62
1869  002376  071522                                    EHT1
1870  002400  071646                                    EDT1
1871  002402  071736                                    EFT1
1872
1873
1874                                    ;ERROR  63      UNUSED
1875  002404  000000                                    0
1876  002406  000000                                    0
1877  002410  000000                                    0
1878  002412  000000                                    0
1879
1880
1881                                    ;ERROR  64      DRIVE DID NOT DETECT "IVC" ERROR DURING SEEK
1882  002414  066264                                    EMT64
1883  002416  071522                                    EHT1
1884  002420  071646                                    EDT1
1885  002422  071736                                    EFT1
1886
1887
1888                                    ;ERROR  65      DRIVE EXECUTED A SEEK WITH ERROR SET
1889  002424  066304                                    EMT65
1890  002426  071522                                    EHT1
1891  002430  071646                                    EDT1
1892  002432  071736                                    EFT1
1893
1894
1895                                    ;ERROR  66      UNEXPECTED ERROR SET IN RMER1
1896  002434  066324                                    EMT66
1897  002436  071522                                    EHT1
1898  002440  071646                                    EDT1
1899  002442  071736                                    EFT1
1900
1901
1902                                    ;ERROR  67      UNEXPECTED ERROR SET IN RMER2
1903  002444  066336                                    EMT67
1904  002446  071522                                    EHT1
1905  002450  071646                                    EDT1
1906  002452  071736                                    EFT1
1907
1908
1909                                    ;ERROR  70      ERRONEOUS "IAE" ERROR DURING RECALIBRATE
1910  002454  066350                                    EMT70
1911  002456  071522                                    EHT1
```

```
1912  002460  071646                          EDT1
1913  002462  071736                          EFT1
1914
1915
1916                                  ;ERROR  71      "ILF" ERROR DURING RECALIBRATE
1917  002464  066360                          EMT71
1918  002466  071522                          EHT1
1919  002470  071646                          EDT1
1920  002472  071736                          EFT1
1921
1922
1923                                  ;ERROR  72      "OPI" ERROR DURING RECALIBRATE DUE TO "MOL" = 0
1924  002474  066370                          EMT72
1925  002476  071522                          EHT1
1926  002500  071646                          EDT1
1927  002502  071736                          EFT1
1928
1929
1930                                  ;ERROR  73      "OPI" ERROR DURING RECALIBRATE BECAUSE ON
1931                                  ;               CYLINDER DIDNT DROP
1932  002504  066406                          EMT73
1933  002506  071522                          EHT1
1934  002510  071646                          EDT1
1935  002512  071736                          EFT1
1936
1937
1938                                  ;ERROR  74      "IVC" ERROR DURING RECALIBRATE - "VV" = 0
1939  002514  066424                          EMT74
1940  002516  071522                          EHT1
1941  002520  071646                          EDT1
1942  002522  071736                          EFT1
1943
1944
1945                                  ;ERROR  75      ERRONEOUS "IVC" ERROR DURING RECALIBRATE - "VV" = 1
1946  002524  066434                          EMT75
1947  002526  071522                          EHT1
1948  002530  071646                          EDT1
1949  002532  071736                          EFT1
1950
1951
1952                                  ;ERROR  76      "SKI" ERROR DURING RECALIBRATE
1953  002534  066454                          EMT76
1954  002536  071522                          EHT1
1955  002540  071646                          EDT1
1956  002542  071736                          EFT1
1957
1958
1959                                  ;ERROR  77      "DVC" OCCURRED DURING RECALIBRATE
1960  002544  066464                          EMT77
1961  002546  071522                          EHT1
1962  002550  071646                          EDT1
1963  002552  071736                          EFT1
1964
1965
1966                                  ;ERROR  100     LOST "MOL" DURING RECALIBRATE - "OPI" = 0
1967  002554  066476                          EMT100
```

```
1968  002556  071522                          EMT1
1969  002560  071646                          EDT1
1970  002562  071736                          EFT1
1971
1972
1973                                  ;ERROR  101     LOST "VV" DURING RECALIBRATE - "IVC" = 0
1974  002564  066514                          EMT101
1975  002566  071522                          EMT1
1976  002570  071646                          EDT1
1977  002572  071736                          EFT1
1978
1979
1980                                  ;ERROR  102     "ATA" DID NOT SET DURING RECALIBRATE
1981  002574  066532                          EMT102
1982  002576  071522                          EMT1
1983  002600  071646                          EDT1
1984  002602  071736                          EFT1
1985
1986
1987                                  ;ERROR  103     "OM" DID NOT RESET DURING RECALIBRATE
1988  002604  066542                          EMT103
1989  002606  071522                          EMT1
1990  002610  071646                          EDT1
1991  002612  071736                          EFT1
1992
1993
1994                                  ;ERROR  104     "PIP" IS STIL SET AFTER RECALIBRATE
1995  002614  066554                          EMT104
1996  002616  071522                          EMT1
1997  002620  071646                          EDT1
1998  002622  071736                          EFT1
1999
2000
2001                                  ;ERROR  105     UNEXPECTED "ILR" ERROR DURING RECALIBRATE
2002  002624  066572                          EMT105
2003  002626  071522                          EMT1
2004  002630  071646                          EDT1
2005  002632  071736                          EFT1
2006
2007
2008                                  ;ERROR  106     UNEXPECTED "RMR" ERROR DURING RECALIBRATE
2009  002634  066602                          EMT106
2010  002636  071522                          EMT1
2011  002640  071646                          EDT1
2012  002642  071736                          EFT1
2013
2014
2015                                  ;ERROR  107     "UNS" ERROR DURING RECALIBRATE - AC POWER IS LOW
2016  002644  066612                          EMT107
2017  002646  071522                          EMT1
2018  002650  071646                          EDT1
2019  002652  071736                          EFT1
2020
2021
2022                                  ;ERROR  110     CANNOT ACCESS MASSBUS CONTROLLER VIA UNIBUS
2023  002654  066632                          EMT110
```

```
2024  002656  071544                    EMT110
2025  002660  071664                    EDT110
2026  002662  071754                    EFT110
2027
2028
2029                              ;ERROR  111     NONEXISTENT DEVICE
2030  002664  066644                    EMT111
2031  002666  071550                    EHT111
2032  002670  071666                    EDT111
2033  002672  071756                    EFT111
2034
2035
2036                              ;ERROR  112     DEVICE NOT AVAILABLE
2037  002674  066652                    EMT112
2038  002676  071550                    EHT111
2039  002700  071666                    EDT111
2040  002702  071756                    EFT111
2041
2042
2043                              ;ERROR  113     BUS TIMEOUT-NED STATUS FAILURE
2044  002704  066660                    EMT113
2045  002706  000000                    0
2046  002710  000000                    0
2047  002712  000000                    0
2048
2049
2050                              ;ERROR  114     DEVICE NOT AN RM03
2051  002714  066674                    EMT114
2052  002716  071554                    EHT114
2053  002720  071670                    EDT114
2054  002722  071760                    EFT114
2055
2056
2057                              ;ERROR  115     RMCS1 NOT INITIALIZED BY UNIBUS
2058  002724  066702                    EMT115
2059  002726  071522                    EHT1
2060  002730  071646                    EDT1
2061  002732  071736                    EFT1
2062
2063
2064                              ;ERROR  116     RMBA NOT INITIALIZED BY UNIBUS
2065  002734  066712                    EMT116
2066  002736  071522                    EHT1
2067  002740  071646                    EDT1
2068  002742  071736                    EFT1
2069
2070
2071                              ;ERROR  117     RMCS2 NOT INITIALIZED BY UNIBUS
2072  002744  066722                    EMT117
2073  002746  071522                    EHT1
2074  002750  071646                    EDT1
2075  002752  071736                    EFT1
2076
2077
2078                              ;ERROR  120     RMER1 NOT INITIALIZED BY UNIBUS
2079  002754  066732                    EMT120
```

```
2080  002756  071522                        EMT1
2081  002760  071646                        EDT1
2082  002762  071736                        EFT1
2083
2084
2085                                ;ERROR  121     RMAS NOT INITIALIZED BY UNIBUS
2086  002764  066742                        EMT121
2087  002766  071522                        EMT1
2088  002770  071646                        EDT1
2089  002772  071736                        EFT1
2090
2091
2092                                ;ERROR  122     RMMR1 NOT INITIALIZED BY UNIBUS
2093  002774  066752                        EMT122
2094  002776  071522                        EMT1
2095  003000  071646                        EDT1
2096  003002  071736                        EFT1
2097
2098
2099                                ;ERROR  123     RMDS NOT INITIALIZED BY UNIBUS
2100  003004  066762                        EMT123
2101  003006  071522                        EMT1
2102  003010  071646                        EDT1
2103  003012  071736                        EFT1
2104
2105
2106                                ;ERROR  124     RMEC2 NOT INITIALIZED BY UNIBUS
2107  003014  066772                        EMT124
2108  003016  071522                        EMT1
2109  003020  071646                        EDT1
2110  003022  071736                        EFT1
2111
2112
2113                                ;ERROR  125     RMMR2 NOT INITIALIZED BY UNIBUS
2114  003024  067002                        EMT125
2115  003026  071522                        EMT1
2116  003030  071646                        EDT1
2117  003032  071736                        EFT1
2118
2119
2120                                ;ERROR  126     RMCS1 NOT CLEARED BY CONTROLLER CLEAR
2121  003034  067012                        EMT126
2122  003036  071522                        EMT1
2123  003040  071646                        EDT1
2124  003042  071736                        EFT1
2125
2126
2127                                ;ERROR  127     RMBA NOT CLEARED BY CONTROLLER CLEAR
2128  003044  067024                        EMT127
2129  003046  071522                        EMT1
2130  003050  071646                        EDT1
2131  003052  071736                        EFT1
2132
2133
2134                                ;ERROR  130     RMCS2 NOT CLEARED BY CONTROLLER CLEAR
2135  003054  067036                        EMT130
```

```
2136  003056  071522                        EHT1
2137  003060  071646                        EDT1
2138  003062  071736                        EFT1
2139
2140
2141                                ;ERROR  131     RMER1 NOT CLEARED BY CONTROLLER CLEAR
2142  003064  067050                        EMT131
2143  003066  071522                        EHT1
2144  003070  071646                        EDT1
2145  003072  071736                        EFT1
2146
2147
2148                                ;ERROR  132     RMAS NOT CLEARED BY CONTROLLER CLEAR
2149  003074  067062                        EMT132
2150  003076  071522                        EHT1
2151  003100  071646                        EDT1
2152  003102  071736                        EFT1
2153
2154
2155                                ;ERROR  133     RMMR1 NOT CLEARED BY CONTROLLER CLEAR
2156  003104  067074                        EMT133
2157  003106  071522                        EHT1
2158  003110  071646                        EDT1
2159  003112  071736                        EFT1
2160
2161
2162                                ;ERROR  134     RMDS NOT CLEARED BY CONTROLLER CLEAR
2163  003114  067106                        EMT134
2164  003116  071522                        EHT1
2165  003120  071646                        EDT1
2166  003122  071736                        EFT1
2167
2168
2169                                ;ERROR  135     RMEC2 NOT CLEARED BY CONTROLLER CLEAR
2170  003124  067120                        EMT135
2171  003126  071522                        EHT1
2172  003130  071646                        EDT1
2173  003132  071736                        EFT1
2174
2175
2176                                ;ERROR  136     RMMR2 NOT CLEARED BY CONTROLLER CLEAR
2177  003134  067132                        EMT136
2178  003136  071522                        EHT1
2179  003140  071646                        EDT1
2180  003142  071736                        EFT1
2181
2182
2183                                ;ERROR  137     RMCS1 NOT CLEARED BY ERROR CLEAR
2184  003144  067144                        EMT137
2185  003146  071522                        EHT1
2186  003150  071646                        EDT1
2187  003152  071736                        EFT1
2188
2189
2190                                ;ERROR  140     RMCS2 NOT CLEARED BY ERROR CLEAR
2191  003154  067154                        EMT140
```

```
2192  003156  071522                        EMT1
2193  003160  071646                        EDT1
2194  003162  071736                        EFT1
2195
2196
2197                                ;ERROR  141     RMCS1 NOT CLEARED BY DRIVE CLEAR
2198  003164  067164                        EMT141
2199  003166  071522                        EMT1
2200  003170  071646                        EDT1
2201  003172  071736                        EFT1
2202
2203
2204                                ;ERROR  142     RMDS NOT CLEARED BY DRIVE CLEAR
2205  003174  067174                        EMT142
2206  003176  071522                        EMT1
2207  003200  071646                        EDT1
2208  003202  071736                        EFT1
2209
2210
2211                                ;ERROR  143     RMER1 NOT CLEARED BY DRIVE CLEAR
2212  003204  067204                        EMT143
2213  003206  071522                        EMT1
2214  003210  071646                        EDT1
2215  003212  071736                        EFT1
2216
2217
2218                                ;ERROR  144     RMAS NOT CLEARED BY DRIVE CLEAR
2219  003214  067214                        EMT144
2220  003216  071522                        EMT1
2221  003220  071646                        EDT1
2222  003222  071736                        EFT1
2223
2224
2225                                ;ERROR  145     RMMR1  NOT CLEARED BY DRIVE CLEAR
2226  003224  067224                        EMT145
2227  003226  071522                        EMT1
2228  003230  071646                        EDT1
2229  003232  071736                        EFT1
2230
2231
2232                                ;ERROR  146     RMMR2 NOT CLEARED BY DRIVE CLEAR
2233  003234  067234                        EMT146
2234  003236  071522                        EMT1
2235  003240  071646                        EDT1
2236  003242  071736                        EFT1
2237
2238
2239                                ;ERROR  147     RMER2 NOT CLEARED BY DRIVE CLEAR
2240  003244  067244                        EMT147
2241  003246  071522                        EMT1
2242  003250  071646                        EDT1
2243  003252  071736                        EFT1
2244
2245
2246                                ;ERROR  150     RMEC2 NOT CLEARED BY DRIVE CLEAR
2247  003254  067254                        EMT150
```

```
2248  003256  071522                        EMT1
2249  003260  071646                        EDT1
2250  003262  071736                        EFT1
2251
2252
2253                                ;ERROR  151     MEDIUM NOT ON LINE
2254  003264  067264                        EMT151
2255  003266  071522                        EMT1
2256  003270  071646                        EDT1
2257  003272  071736                        EFT1
2258
2259
2260                                ;ERROR  152     DRIVE FAULT
2261  003274  067276                        EMT152
2262  003276  071522                        EMT1
2263  003300  071646                        EDT1
2264  003302  071736                        EFT1
2265
2266
2267                                ;ERROR  153     UNSAFE SHOULD BE SET BECAUSE DVC IS SET
2268  003304  067310                        EMT153
2269  003306  071522                        EMT1
2270  003310  071646                        EDT1
2271  003312  071736                        EFT1
2272
2273
2274                                ;ERROR  154     UNSAFE SHOULD NOT BE SET, AC IS LOW
2275  003314  067326                        EMT154
2276  003316  071522                        EMT1
2277  003320  071646                        EDT1
2278  003322  071736                        EFT1
2279
2280
2281                                ;ERROR  155     VOLUME VALID NOT SET BY PACK ACK
2282  003324  067344                        EMT155
2283  003326  071522                        EMT1
2284  003330  071646                        EDT1
2285  003332  071736                        EFT1
2286
2287
2288                                ;ERROR  156     OFFSET MODE NOT SET BY OFFSET COMMAND
2289  003334  067356                        EMT156
2290  003336  071522                        EMT1
2291  003340  071646                        EDT1
2292  003342  071736                        EFT1
2293
2294
2295                                ;ERROR  157     OFFSET MODE NOT RESET BY RTC COMMAND
2296  003344  067370                        EMT157
2297  003346  071522                        EMT1
2298  003350  071646                        EDT1
2299  003352  071736                        EFT1
2300
2301
2302                                ;ERROR  160     RMOF NOT RESET BY RIP COMMAND
2303  003354  067402                        EMT160
```

```
2304  003356  071522                        EMT1
2305  003360  071646                        EDT1
2306  003362  071736                        EFT1
2307
2308
2309                                ;ERROR  161     RMDA NOT RESET BY RIP COMMAND
2310  003364  067412                        EMT161
2311  003366  071522                        EMT1
2312  003370  071646                        EDT1
2313  003372  071736                        EFT1
2314
2315
2316                                ;ERROR  162     RMDC NOT RESET BY RIP COMMAND
2317  003374  067424                        EMT162
2318  003376  071522                        EMT1
2319  003400  071646                        EDT1
2320  003402  071736                        EFT1
2321
2322
2323                                ;ERROR 163      DATA WAS ECC CORRECTED BUT DOES NOT COMPARE WITH
2324                                ;               WRITE BUFFER
2325  003404  071316                        EMT336
2326  003406  071604                        EMT336
2327  003410  071702                        EDT336
2328  003412  071772                        EFT336
2329
2330
2331                                ;ERROR  164     OPI SHOULD NOT BE SET
2332  003414  067446                        EMT164
2333  003416  071522                        EMT1
2334  003420  071646                        EDT1
2335  003422  071736                        EFT1
2336
2337
2338                                ;ERROR  165     IVC SHOULD NOT BE SET
2339  003424  067454                        EMT165
2340  003426  071522                        EMT1
2341  003430  071646                        EDT1
2342  003432  071736                        EFT1
2343
2344
2345                                ;ERROR  166     IAE SHOULD NOT BE SET
2346  003434  067462                        EMT166
2347  003436  071522                        EMT1
2348  003440  071646                        EDT1
2349  003442  071736                        EFT1
2350
2351
2352                                ;ERROR  167     NEM SHOULD NOT BE SET
2353  003444  067470                        EMT167
2354  003446  071522                        EMT1
2355  003450  071646                        EDT1
2356  003452  071736                        EFT1
2357
2358
2359                                ;ERROR 170      UNUSED
```

```
2360  003454  000000                    0
2361  003456  000000                    0
2362  003460  000000                    0
2363  003462  000000                    0
2364
2365
2366                              ;ERROR  171     "ATA" NOT SET DURING RETURN TO CENTERLINE
2367  003464  067506                      EMT171
2368  003466  071522                      EHT1
2369  00347C  071646                      EDT1
2370  003472  071736                      EFT1
2371
2372
2373                              ;ERROR  172     "ATA" NOT SET BY OFFSET COMMAND
2374  003474  067516                      EMT172
2375  003476  071522                      EHT1
2376  003500  071646                      EDT1
2377  003502  071736                      EFT1
2378
2379
2380                              ;ERROR  173     RMER2 NOT INITIALIZED BY UNIBUS INIT
2381  003504  067526                      EMT173
2382  003506  071522                      EHT1
2383  003510  071646                      EDT1
2384  003512  071736                      EFT1
2385
2386
2387                              ;ERROR  174     RMER2 NOT INITIALIZED BY CONTROLLER CLEAR
2388  003514  067536                      EMT174
2389  003516  071522                      EHT1
2390  003520  071646                      EDT1
2391  003522  071736                      EFT1
2392
2393
2394                              ;ERROR  175     SELECTED DEVICE IS IN WRITE PROTECT
2395  003524  067550                      EMT175
2396  003526  071522                      EHT1
2397  003530  071646                      EDT1
2398  003532  071736                      EFT1
2399
2400
2401                              ;ERROR  176     CANNOT SET DIAGNOSTIC MODE
2402  003534  067556                      EMT176
2403  003536  071522                      EHT1
2404  003540  071646                      EDT1
2405  003542  071736                      EFT1
2406
2407
2408                              ;ERROR  177     INCORRECT "MOL" STATUS DURING DIAGNOSTIC MODE
2409  003544  067564                      EMT177
2410  003546  071522                      EHT1
2411  003550  071646                      EDT1
2412  003552  071736                      EFT1
2413
2414
2415                              ;ERROR  200     INCORRECT "PIP" STATUS DURING DIAGNOSTIC MODE
```

```
2416  003554  067576                    EMT200
2417  003556  071522                    EHT1
2418  003560  071646                    EDT1
2419  003562  071736                    EFT1
2420
2421
2422                              ;ERROR  201     INCORRECT "WRL" STATUS DURING DIAGNOSTIC MODE
2423  003564  067610                    EMT201
2424  003566  071522                    EHT1
2425  003570  071646                    EDT1
2426  003572  071736                    EFT1
2427
2428
2429                              ;ERROR  202     INCORRECT "SKI" STATUS DURING DIAGNOSTIC MODE
2430  003574  067622                    EMT202
2431  003576  071522                    EHT1
2432  003600  071646                    EDT1
2433  003602  071736                    EFT1
2434
2435
2436                              ;ERROR  203     INCORRECT "DVC" STATUS DURING DIAGNOSTIC MODE
2437  003604  067634                    EMT203
2438  003606  071522                    EHT1
2439  003610  071646                    EDT1
2440  003612  071736                    EFT1
2441
2442
2443                              ;ERROR  204     "VV" WAS NOT RESET BY MAINTENANCE UNIT READY
2444  003614  067646                    EMT204
2445  003616  071522                    EHT1
2446  003620  071646                    EDT1
2447  003622  071736                    EFT1
2448
2449
2450                              ;ERROR  205     SELECTED DEVICE HAS A PERSISTENT "SKI" ERROR
2451  003624  067664                    EMT205
2452  003626  071522                    EHT1
2453  003630  071646                    EDT1
2454  003632  071736                    EFT1
2455
2456
2457                              ;ERROR  206     "LBC" DID NOT SET DURING DIAGNOSTIC MODE
2458  003634  067674                    EMT206
2459  003636  071522                    EHT1
2460  003640  071646                    EDT1
2461  003642  071736                    EFT1
2462
2463
2464                              ;ERROR  207     UNEXPECTED LOSS OF "MOL" - MEDIUM IS OFF LINE
2465  003644  067704                    EMT207
2466  003646  071522                    EHT1
2467  003650  071646                    EDT1
2468  003652  071736                    EFT1
2469
2470
2471                              ;ERROR  210     UNEXPECTED LOSS OF VOLUME VALID - "VV" = 0
```

```
2472  003654  067716                    EMT210
2473  003656  071522                    EHT1
2474  003660  071646                    EDT1
2475  003662  071736                    EFT1
2476
2477
2478                              ;ERROR  211     UNEXPECTED MECHANICAL MOTION - "PIP" = 1
2479  003664  067724                    EMT211
2480  003666  071522                    EHT1
2481  003670  071646                    EDT1
2482  003672  071736                    EFT1
2483
2484
2485                              ;ERROR  212     UNEXPECTED DEVICE FAULT - "DVC" = 1
2486  003674  067740                    EMT212
2487  003676  071522                    EHT1
2488  003700  071646                    EDT1
2489  003702  071736                    EFT1
2490
2491
2492                              ;ERROR  213     UNEXPECTED SEEK INCOMPLETE ERROR - "SKI" = 1
2493  003704  067754                    EMT213
2494  003706  071522                    EHT1
2495  003710  071646                    EDT1
2496  003712  071736                    EFT1
2497
2498
2499                              ;ERROR  214     DRIVE EXECUTED A RECALIBRATE WITH ERROR SET
2500  003714  067764                    EMT214
2501  003716  071534                    EHT2
2502  003720  071656                    EDT2
2503  003722  071746                    EFT2
2504
2505
2506                              ;ERROR  215     DRIVE DID NOT DETECT "IVC" ERROR DURING RECALIBRATE
2507  003724  070004                    EMT215
2508  003726  071534                    EHT2
2509  003730  071656                    EDT2
2510  003732  071746                    EFT2
2511
2512
2513                              ;ERROR  216     INCORRECT "IVC" STATUS
2514  003734  070016                    EMT216
2515  003736  071522                    EHT1
2516  003740  071646                    EDT1
2517  003742  071736                    EFT1
2518
2519
2520                              ;ERROR  217     INCORRECT "IAE" STATUS
2521  003744  070026                    EMT217
2522  003746  071522                    EHT1
2523  003750  071646                    EDT1
2524  003752  071736                    EFT1
2525
2526
2527                              ;ERROR 220      INCORRECT "WLE" STATUS
```

```
2528  003754  070036                    EMT220
2529  003756  071522                    EHT1
2530  003760  071646                    EDT1
2531  003762  071736                    EFT1
2532
2533
2534                              ;ERROR  221     INCORRECT "OPI" STATUS
2535  003764  070046                    EMT221
2536  003766  071522                    EHT1
2537  003770  071646                    EDT1
2538  003772  071736                    EFT1
2539
2540
2541                              ;ERROR  222     RM03 DID NOT DETECT RMR ERROR
2542  003774  070056                    EMT222
2543  003776  071522                    EHT1
2544  004000  071646                    EDT1
2545  004002  071736                    EFT1
2546
2547
2548                              ;ERROR  223     RM03 DID NOT DETECT PARITY ERROR ON MASSBUS CONTROL BUS
2549  004004  070066                    EMT223
2550  004006  071560                    EHT223
2551  004010  071672                    EDT223
2552  004012  071762                    EFT223
2553
2554
2555                              ;ERROR  224     UNUSED
2556  004014  000000                    0
2557  004016  000000                    0
2558  004020  000000                    0
2559  004022  000000                    0
2560
2561
2562                              ;ERROR  225     UNUSED
2563  004024  000000                    0
2564  004026  000000                    0
2565  004030  000000                    0
2566  004032  000000                    0
2567
2568
2569                              ;ERROR  226     UNUSED
2570  004034  000000                    0
2571  004036  000000                    0
2572  004040  000000                    0
2573  004042  000000                    0
2574
2575
2576                              ;ERROR  227     UNUSED
2577  004044  000000                    0
2578  004046  000000                    0
2579  004050  000000                    0
2580  004052  000000                    0
2581
2582
2583                              ;ERROR  230     UNUSED
```

```
2584  004054  000000                    0
2585  004056  000000                    0
2586  004060  000000                    0
2587  004062  000000                    0
2588
2589
2590                              ;ERROR  231     UNUSED
2591  004064  000000                    0
2592  004066  000000                    0
2593  004070  000000                    0
2594  004072  000000                    0
2595
2596
2597                              ;ERROR  232     UNUSED
2598  004074  000000                    0
2599  004076  000000                    0
2600  004100  000000                    0
2601  004102  000000                    0
2602
2603
2604                              ;ERROR  233     UNUSED
2605  004104  000000                    0
2606  004106  000000                    0
2607  004110  000000                    0
2608  004112  000000                    0
2609
2610
2611                              ;ERROR  234     UNUSED
2612  004114  000000                    0
2613  004116  000000                    0
2614  004120  000000                    0
2615  004122  000000                    0
2616
2617
2618                              ;ERROR  235     UNUSED
2619  004124  000000                    0
2620  004126  000000                    0
2621  004130  000000                    0
2622  004132  000000                    0
2623
2624
2625                              ;ERROR  236     UNUSED
2626  004134  000000                    0
2627  004136  000000                    0
2628  004140  000000                    0
2629  004142  000000                    0
2630
2631
2632                              ;ERROR  237     UNUSED
2633  004144  000000                    0
2634  004146  000000                    0
2635  004150  000000                    0
2636  004152  000000                    0
2637
2638
2639                              ;ERROR  240     UNUSED
```

```
2640  004154  000000                    0
2641  004156  000000                    0
2642  004160  000000                    0
2643  004162  000000                    0
2644
2645
2646                              ;ERROR  241     UNUSED
2647  004164  000000                    0
2648  004166  000000                    0
2649  004170  000000                    0
2650  004172  000000                    0
2651
2652
2653                              ;ERROR  242     UNUSED
2654  004174  000000                    0
2655  004176  000000                    0
2656  004200  000000                    0
2657  004202  000000                    0
2658
2659
2660                              ;ERROR  243     UNUSED
2661  004204  000000                    0
2662  004206  000000                    0
2663  004210  000000                    0
2664  004212  000000                    0
2665
2666
2667                              ;ERROR  244     UNUSED
2668  004214  000000                    0
2669  004216  000000                    0
2670  004220  000000                    0
2671  004222  000000                    0
2672
2673
2674                              ;ERROR  245     UNUSED
2675  004224  000000                    0
2676  004226  000000                    0
2677  004230  000000                    0
2678  004232  000000                    0
2679
2680
2681                              ;ERROR  246     "ATA" NOT RESET BY GO WHEN "ERR" = 0
2682  004234  070142                    EMT246
2683  004236  071522                    EHT1
2684  004240  071646                    EDT1
2685  004242  071736                    EFT1
2686
2687
2688                              ;ERROR  247     "ATA" NOT RESET BY WRITING RMAS
2689  004244  070152                    EMT247
2690  004246  071522                    EHT1
2691  004250  071646                    EDT1
2692  004252  071736                    EFT1
2693
2694
2695                              ;ERROR  250     "ATA" WAS RESET BY GO WHEN "ERR" = 1
```

```
2696  004254  070164                        EMT250
2697  004256  071522                        EHT1
2698  004260  071646                        EDT1
2699  004262  071736                        EFT1
2700
2701
2702                                ;ERROR  251     PROGRAM INTERRUPT WAS NOT GENERATED
2703  004264  070200                        EMT251
2704  004266  071534                        EHT2
2705  004270  071656                        EDT2
2706  004272  071746                        EFT2
2707
2708
2709                                ;ERROR  252     PROGRAM INTERRUPT SHOULD NOT HAVE BEEN GENERATED
2710  004274  070206                        EMT252
2711  004276  071534                        EHT2
2712  004300  071656                        EDT2
2713  004302  071746                        EFT2
2714
2715
2716                                ;ERROR  253     OFFSET MODE WAS NOT RESET BY WRITING RMDC
2717  004304  070214                        EMT253
2718  004306  071522                        EHT1
2719  004310  071646                        EDT1
2720  004312  071736                        EFT1
2721
2722
2723                                ;ERROR  254     INCORRECT "ILF" STATUS
2724  004314  070232                        EMT254
2725  004316  071522                        EHT1
2726  004320  071646                        EDT1
2727  004322  071736                        EFT1
2728
2729
2730                                ;ERROR  255     INCORRECT "ATA" STATUS
2731  004324  070242                        EMT255
2732  004326  071522                        EHT1
2733  004330  071646                        EDT1
2734  004332  071736                        EFT1
2735
2736
2737                                ;ERROR  256     INCORRECT "ILR" STATUS
2738  004334  070252                        EMT256
2739  004336  071572                        EHT256
2740  004340  071672                        EDT223
2741  004342  071762                        EFT223
2742
2743
2744                                ;ERROR  257     INVALID IAE STATUS DURING SEARCH COMMAND
2745  004344  070262                        EMT257
2746  004346  071522                        EHT1
2747  004350  071646                        EDT1
2748  004352  071736                        EFT1
2749
2750
2751                                ;ERROR  260     "IVC" WAS NOT DETECTED DURING SEARCH COMMAND
```

```
2752  004354  070274                    EMT260
2753  004356  071522                    EHT1
2754  004360  071646                    EDT1
2755  004362  071736                    EFT1
2756
2757
2758                              ;ERROR  261     DRIVE EXECUTED SEARCH WITH ERROR SET
2759  004364  070306                    EMT261
2760  004366  071522                    EHT1
2761  00437C  071646                    EDT1
2762  004372  071736                    EFT1
2763
2764
2765                              ;ERROR  262     "LBC" ERROR NOT SET DURING DIAGNOSTIC MODE
2766  004374  070326                    EMT262
2767  004376  071522                    EHT1
2768  004400  071646                    EDT1
2769  004402  071736                    EFT1
2770
2771
2772                              ;ERROR  263     "SKI" ERROR DURING SEARCH COMMAND
2773  004404  070336                    EMT263
2774  004406  071522                    EHT1
2775  004410  071646                    EDT1
2776  004412  071736                    EFT1
2777
2778
2779                              ;ERROR  264     "IVC" ERROR DURING SEARCH - LOST VOLUME VALID
2780  004414  070346                    EMT264
2781  004416  071522                    EHT1
2782  004420  071646                    EDT1
2783  004422  071736                    EFT1
2784
2785
2786                              ;ERROR  265     ERRONEOUS IVC ERROR DURING SEARCH - VOLUME IS VALID
2787  004424  070366                    EMT265
2788  004426  071522                    EHT1
2789  004430  071646                    EDT1
2790  004432  071736                    EFT1
2791
2792
2793                              ;ERROR  266     DEVICE FAULT (DVC) DURING SEARCH
2794  004434  070406                    EMT266
2795  004436  071522                    EHT1
2796  004440  071646                    EDT1
2797  004442  071736                    EFT1
2798
2799
2800                              ;ERROR  267     SKI SHOULD HAVE BEEN SET BECAUSE CYLINDER
2801                              ;               ADDRESS IS TOO LARGE
2802  004444  070420                    EMT267
2803  004446  071522                    EHT1
2804  004450  071646                    EDT1
2805  004452  071736                    EFT1
2806
2807
```

```
2808
2809  004454  070436                    ;ERROR  270     OPI ERROR DURING SEARCH BECAUSE MOL = 0
2810  004456  071522                            EMT270
2811  004460  071646                            EHT1
2812  004462  071736                            EDT1
2813                                      EFT1
2814
2815                                      ;ERROR  271     OPI ERROR DURING SEARCH BECAUSE ON CYLINDER
2816                                      ;               DIDN'T DROP
2817  004464  070452                            EMT271
2818  004466  071522                            EHT1
2819  004470  071646                            EDT1
2820  004472  071736                            EFT1
2821
2822
2823                                      ;ERROR  272     LOST MOL DURING SEARCH, OPI IS NOT SET
2824  004474  070470                            EMT272
2825  004476  071522                            EHT1
2826  004500  071646                            EDT1
2827  004502  071736                            EFT1
2828
2829
2830                                      ;ERROR  273     PIP STIL SET AFTER SEARCH
2831  004504  070506                            EMT273
2832  004506  071522                            EHT1
2833  004510  071646                            EDT1
2834  004512  071736                            EFT1
2835
2836
2837                                      ;ERROR  274     PARITY ERROR OCCURRED WHILE WRITING REMOTE
2838                                      ;               REGISTERS BUT MXF DID NOT SET
2839  004514  070524                            EMT274
2840  004516  071522                            EHT1
2841  004520  071646                            EDT1
2842  004522  071736                            EFT1
2843
2844
2845                                      ;ERROR  275     MXF ERROR - COMPOSITE ERROR OCCURRED BEFORE DATA
2846                                      ;               COMMAND STARTED
2847  004524  070542                            EMT275
2848  004526  071522                            EHT1
2849  004530  071646                            EDT1
2850  004532  071736                            EFT1
2851
2852
2853                                      ;ERROR  276     "OPI" ERROR DURING DATA TRANSFER BECAUSE "MOL" WAS
2854                                      ;               ZERO
2855  004534  070554                            EMT276
2856  004536  071522                            EHT1
2857  004540  071646                            EDT1
2858  004542  071736                            EFT1
2859
2860
2861                                      ;ERROR  277     "OPI" ERROR DURING DATA TRANSFER BECAUSE 1) ON
2862                                      ;               CYLINDER DIDN'T DROP OR 2) SEARCH TIMED OUT OR
2863                                      ;               3) RUN TIMED OUT
```

```
2864  004544  070570                    EMT277
2865  004546  071522                    EHT1
2866  004550  071646                    EDT1
2867  004552  071736                    EFT1
2868
2869
2870                          ;ERROR  300     "IVC" ERROR DURING DATA TRANSFER BECAUSE VOLUME
2871                          ;               WAS NOT VALID
2872  004554  070606                    EMT300
2873  004556  071522                    EHT1
2874  004560  071646                    EDT1
2875  004562  071736                    EFT1
2876
2877
2878                          ;ERROR  301     ERRONEOUS "IVC" ERROR DURING DATA TRANSFER - VOLUME
2879                          ;               IS VALID
2880  004564  070626                    EMT301
2881  004566  071522                    EHT1
2882  004570  071646                    EDT1
2883  004572  071736                    EFT1
2884
2885
2886                          ;ERROR  302     "ILR" ERROR DURING DATA TRANSFER
2887  004574  070650                    EMT302
2888  004576  071522                    EHT1
2889  004600  071646                    EDT1
2890  004602  071736                    EFT1
2891
2892
2893                          ;ERROR  303     "ILF" ERROR DURING DATA TRANSFER
2894  004604  070662                    EMT303
2895  004606  071522                    EHT1
2896  004610  071646                    EDT1
2897  004612  071736                    EFT1
2898
2899
2900                          ;ERROR  304     "RMR" ERROR DURING DATA TRANSFER
2901  004614  070674                    EMT304
2902  004616  071522                    EHT1
2903  004620  071646                    EDT1
2904  004622  071736                    EFT1
2905
2906
2907                          ;ERROR  305     INCORRECT "IAE" STATUS DURING DATA TRANSFER
2908  004624  070706                    EMT305
2909  004626  071522                    EHT1
2910  004630  071646                    EDT1
2911  004632  071736                    EFT1
2912
2913
2914                          ;ERROR  306     "SKI" ERROR DURING DATA TRANSFER
2915  004634  070720                    EMT306
2916  004636  071522                    EHT1
2917  004640  071646                    EDT1
2918  004642  071736                    EFT1
2919
```

```
2920
2921                                    ;ERROR  307     DRIVE DID NOT DETECT SKI ERROR DUE TO CYLINDER
2922  004644  070730                              EMT307
2923  004646  071522                              EHT1
2924  004650  071646                              EDT1
2925  004652  071736                              EFT1
2926
2927
2928                                    ;ERROR  310     DEVICE FAULT DURING DATA TRANSFER
2929  004654  070750                              EMT310
2930  004656  071522                              EHT1
2931  004660  071646                              EDT1
2932  004662  071736                              EFT1
2933
2934
2935                                    ;ERROR  311     LOSS OF BIT CLOCK DURING DATA TRANSFER
2936  004664  070762                              EMT311
2937  004666  071522                              EHT1
2938  004670  071646                              EDT1
2939  004672  071736                              EFT1
2940
2941
2942                                    ;ERROR  312     LOSS OF SYSTEM CLOCK DURING DATA TRANSFER
2943  004674  070774                              EMT312
2944  004676  071522                              EHT1
2945  004700  071646                              EDT1
2946  004702  071736                              EFT1
2947
2948
2949                                    ;ERROR  313     UNSAFE ERROR DURING DATA TRANSFER (DVC = 0)
2950  004704  071006                              EMT313
2951  004706  071522                              EHT1
2952  004710  071646                              EDT1
2953  004712  071736                              EFT1
2954
2955
2956                                    ;ERROR  314     DRIVE TIMING ERROR DURING DATA TRANSFER
2957  004714  071026                              EMT314
2958  004716  071522                              EHT1
2959  004720  071646                              EDT1
2960  004722  071736                              EFT1
2961
2962
2963                                    ;ERROR  315     WRITE LOCK ERROR
2964  004724  071040                              EMT315
2965  004726  071522                              EHT1
2966  004730  071646                              EDT1
2967  004732  071736                              EFT1
2968
2969
2970                                    ;ERROR  316     ERRONEOUS WRITE LOCK ERROR
2971  004734  071052                              EMT316
2972  004736  071522                              EHT1
2973  004740  071646                              EDT1
2974  004742  071736                              EFT1
2975
```

```
2976
2977                                    ;ERROR  317     HEADER CRC ERROR DURING DATA TRANSFER
2978  004744  071064                              EMT317
2979  004746  071522                              EHT1
2980  004750  071646                              EDT1
2981  004752  071736                              EFT1
2982
2983
2984                                    ;ERROR  320     FORMAT ERROR DURING DATA TRANSFER
2985  004754  071074                              EMT320
2986  004756  071522                              EHT1
2987  004760  071646                              EDT1
2988  004762  071736                              EFT1
2989
2990
2991                                    ;ERROR  321     HEADER COMPARE ERROR DURING DATA TRANSFER
2992  004764  071104                              EMT321
2993  004766  071522                              EHT1
2994  004770  071646                              EDT1
2995  004772  071736                              EFT1
2996
2997
2998                                    ;ERROR  322     HEADER ERRORS SHOULD NOT BE SET
2999  004774  071114                              EMT322
3000  004776  071522                              EHT1
3001  005000  071646                              EDT1
3002  005002  071736                              EFT1
3003
3004
3005                                    ;ERROR  323     DATA CHECK ERROR DURING DATA TRANSFER
3006  005004  071122                              EMT323
3007  005006  071522                              EHT1
3008  005010  071646                              EDT1
3009  005012  071736                              EFT1
3010
3011
3012                                    ;ERROR  324     CORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
3013  005014  071132                              EMT324
3014  005016  071522                              EHT1
3015  005020  071646                              EDT1
3016  005022  071736                              EFT1
3017
3018
3019                                    ;ERROR  325     UNCORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
3020  005024  071144                              EMT325
3021  005026  071522                              EHT1
3022  005030  071646                              EDT1
3023  005032  071736                              EFT1
3024
3025
3026                                    ;ERROR  326     DATA PARITY ERROR DURING READ COMMMAND
3027  005034  071156                              EMT326
3028  005036  071522                              EHT1
3029  005040  071646                              EDT1
3030  005042  071736                              EFT1
3031
```

```
3032
3033                                    ;ERROR  327     OFFSET MODE NOT RESET BY WRITE COMMAND
3034  005044  071174                              EMT327
3035  005046  071522                              EHT1
3036  005050  071646                              EDT1
3037  005052  071736                              EFT1
3038
3039
3040                                    ;ERROR  330     DATA PARITY ERROR DURING WRITE COMMAND
3041  005054  071206                              EMT330
3042  005056  071522                              EHT1
3043  005060  071646                              EDT1
3044  005062  071736                              EFT1
3045
3046
3047                                    ;ERROR  331     WRITE CLOCK FALURE DURING WRITE COMMAND
3048  005064  071216                              EMT331
3049  005066  071522                              EHT1
3050  005070  071646                              EDT1
3051  005072  071736                              EFT1
3052
3053
3054                                    ;ERROR  332     DATA LATE ERROR DURING DATA TRANSFER
3055  005074  071230                              EMT332
3056  005076  071522                              EHT1
3057  005100  071646                              EDT1
3058  005102  071736                              EFT1
3059
3060
3061                                    ;ERROR  333     PIP STIL SET AFTER DATA TRANSFER - SKI = 0
3062  005104  071242                              EMT333
3063  005106  071522                              EHT1
3064  005110  071646                              EDT1
3065  005112  071736                              EFT1
3066
3067
3068                                    ;ERROR  334     LOST MOL DURING DATA TRANSFER - OPI = 0
3069  005114  071260                              EMT334
3070  005116  071522                              EHT1
3071  005120  071646                              EDT1
3072  005122  071736                              EFT1
3073
3074
3075                                    ;ERROR  335     LOST VOLUME VALID DURING DATA TRANSFER - IVC  = 0
3076  005124  071276                              EMT335
3077  005126  071522                              EHT1
3078  005130  071646                              EDT1
3079  005132  071736                              EFT1
3080
3081
3082                                    ;ERROR  336     DATA READ DOES NOT COMPARE WITH DATA WRITTEN
3083  005134  071316                              EMT336
3084  005136  071604                              EHT336
3085  005140  071702                              EDT336
3086  005142  071772                              EFT336
3087
```

```
3088
3089                                    ;ERROR  337     WRITE CHECK ERROR NOT DETECTED
3090  005144  071326                              EMT337
3091  005146  071616                              EHT337
3092  005150  071712                              EDT337
3093  005152  072002                              EFT337
3094
3095
3096                                    ;ERROR  340     WRITE CHECK ERROR AT UNEXPECTED ADDRESS
3097  005154  071336                              EMT340
3098  005156  071604                              EHT336
3099  005160  071702                              EDT336
3100  005162  071772                              EFT336
3101
3102
3103                                    ;ERROR  341     INCORRECT DATA DURING WRITE CHECK ERROR
3104  005164  071350                              EMT341
3105  005166  071604                              EHT336
3106  005170  071702                              EDT336
3107  005172  071772                              EFT336
3108
3109
3110                                    ;ERROR  342     "IVC" ERROR NOT DETECTED DURING DATA TRANSFER
3111  005174  071356                              EMT342
3112  005176  071522                              EHT1
3113  005200  071646                              EDT1
3114  005202  071736                              EFT1
3115
3116
3117                                    ;ERROR  343     "FER" NOT DETECTED DURING DATA TRANSFER
3118  005204  071370                              EMT343
3119  005206  071522                              EHT1
3120  005210  071646                              EDT1
3121  005212  071736                              EFT1
3122
3123
3124                                    ;ERROR  344     "HCE" NOT DETECTED DURING DATA TRANSFER
3125  005214  071402                              EMT344
3126  005216  071630                              EHT344
3127  005220  071722                              EDT344
3128  005222  072012                              EFT344
3129
3130
3131                                    ;ERROR  345     "BSE" NOT DETECTED DURING DATA TRANSFER
3132  005224  071414                              EMT345
3133  005226  071522                              EHT1
3134  005230  071646                              EDT1
3135  005232  071736                              EFT1
3136
3137
3138                                    ;ERROR  346     HEADER ERROR WAS DETECTED W/ HCI SET
3139  005234  071424                              EMT346
3140  005236  071522                              EHT1
3141  005240  071646                              EDT1
3142  005242  071736                              EFT1
3143
```

```
3144
3145                                    ;ERROR  347     DATA TRANSFER NOT ABORTED W/ COMP ERROR SET
3146  005244  071440                              EMT347
3147  005246  071522                              EHT1
3148  005250  071646                              EDT1
3149  005252  071736                              EFT1
3150
3151
3152                                    ;ERROR  350     LOST VOLUME VALID DURING SEARCH - "IVC" = 0
3153  005254  071452                              EMT350
3154  005256  071522                              EHT1
3155  005260  071646                              EDT1
3156  005262  071736                              EFT1
3157
3158
3159                                    ;ERROR  351     "ATA" DID NOT SET DURING SEARCH
3160  005264  071470                              EMT351
3161  005266  071522                              EHT1
3162  005270  071646                              EDT1
3163  005272  071736                              EFT1
3164
3165
3166                                    ;ERROR  352     PROGRAM TIMEOUT WHILE TESTING RMLA
3167  005274  071500                              EMT352
3168  005276  000000                              0
3169  005300  000000                              0
3170  005302  000000                              0
3171
3172
3173                                    ;ERROR  353     LOOK AHEAD TEST FAILS
3174  005304  071504                              EMT353
3175  005306  071642                              EHT353
3176  005310  071734                              EDT353
3177  005312  072022                              EFT353
3178
3179
3180                                    ;ERROR  354     BSE SHOULD NOT BE SET
3181  005314  071514                              EMT354
3182  005316  071522                              EHT1
3183  005320  071646                              EDT1
3184  005322  071736                              EFT1
3185
3186
3187                                    ;PUT ERROR TABLE HERE
3188                                    .SBTTL  ERROR TABLE USAGE
3189
3190                                    ;THE ERROR TABLE ABOVE CONSISTS OF FOUR WORD ENTRIES FOR EACH ERROR
3191                                    ;NUMBER, I.E.,
3192                                    ;
3193                                    ;       EMT - ERROR MESSAGE TABLE ADDRESS
3194                                    ;       EHT - ERROR HEADER TABLE ADDRESS
3195                                    ;       EDT - ERROR DATA TABLE ADDRESS
3196                                    ;       EFT - ERROR FORMAT TABLE ADDRESS
3197                                    ;
3198                                    ;THE EMT ENTRY IS THE ADDRESS OF THE TABLE OF ERROR MESSAGE STRINGS
3199                                    ;FOR THE PARTICULAR ERROR.  EACH ERROR MESSAGE TABLE LISTS THE ADDRESS
```

```
3200                    ;OF ONE OR MORE ERROR MESSAGE STRINGS WHICH ARE TO BE FORMATTED AND
3201                    ;TYPED BY THE ERROR TYPE SUBROUTINE.  IF THE EMT ENTRY IS ZERO, THERE IS
3202                    ;NO MESSAGE TO BE TYPED FOR THE ERROR.
3203                    ;
3204                    ;SIMILARLY, THE EHT, EDT, AND EFT ENTRIES ARE ADDRESSES OF TABLES
3205                    ;OF HEADER, DATA AND FORMAT INFORMATION FOR A GIVEN ERROR.  EACH ENTRY
3206                    ;IN THE ERROR HEADER TABLE MAY OR MAY NOT HAVE AN ASSOCIATED LINE OF
3207                    ;DATA, HOWEVER, EACH DATA LINE MUST HAVE AN ASSOCIATED FORMAT AND
3208                    ;HEADER. THAT IS, A HEADER LINE MAY BE PRINTED WITHOUT ANY DATA,
3209                    ;BUT A DATA LINE IS NOT PRINTED WITHOUT A HEADER, AND EACH DATA LINE
3210                    ;MUST ALSO HAVE A FORMAT.
3211                    ;
3212                    ;IN SUMMARY,
3213                    ;
3214                    ;       EACH NONZERO ENTRY IS THE ADDRESS OF A TABLE,
3215                    ;       EACH TABLE IS A LIST OF ADDRESSES WHICH DEFINES THE LOCATIONS
3216                    ;OF MESSAGE STRINGS, HEADERS, DATA OR FORMAT.
3217                    ;
3218
```

```
3219  .                                 .SBTTL  START OF PROGRAM
3220
3221
3222  005324                            START:
3223
3224                                    ;CLEAR AND SETUP FOR TEST EXECUTION
3225  005324  000240                           NOP
3226  005326  000005                           RESET                   ;INITIALIZE THE SYSTEM
3227  005330  013746  000300                   MOV     PR6,-(SP)       ;;PUT NEW PS ON STACK
3228  005334  012746  005342                   MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
3229  005340  000002                           RTI                     ;;POP NEW PC AND PS
3230  005342                            64$:
3231
3232                                    .SBTTL  INITIALIZE THE COMMON TAGS
3233                                    ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
3234  005342  012706  001114                   MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
3235  005346  005026                           CLR     (R6)+           ;;CLEAR MEMORY LOCATION
3236  005350  022706  001154                   CMP     #SWR,R6 ;;DONE?
3237  005354  001374                           BNE     .-6             ;;LOOP BACK IF NO
3238  005356  012706  001100                   MOV     #STACK,SP       ;;SETUP THE STACK POINTER
3239                                    ;;INITIALIZE A FEW VECTORS
3240  005362  012737  060556  000020           MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
3241  005370  012737  000340  000022           MOV     #340,@#IOTVEC+2 ;;LEVEL 7
3242  005376  012737  061166  000030           MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
3243  005404  012737  000340  000032           MOV     #340,@#EMTVEC+2 ;;LEVEL 7
3244  005412  012737  062726  000034           MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
3245  005420  012737  000340  000036           MOV     #340,@#TRAPVEC+2;LEVEL 7
3246  005426  012737  063034  000024           MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
3247  005434  012737  000340  000026           MOV     #340,@#PWRVEC+2 ;;LEVEL 7
3248  005442  013737  033022  033014           MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
3249  005450  005037  001206                   CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
3250  005454  005037  001210                   CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
3251  005460  112737  000001  001131           MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
3252  005466  012737  005466  001122           MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
3253  005474  012737  005474  001124           MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
3254                                    ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
3255                                    ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
3256  005502  013746  000004                   MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
3257  005506  012737  005542  000004           MOV     #65$,@#ERRVEC   ;;SET UP ERROR VECTOR
3258  005514  012737  177570  001154           MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
3259  005522  012737  177570  001156           MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
3260  005530  022777  177777  173416           CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
3261  005536  001012                           BNE     67$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
3262                                                                    ;;AND  THE HARDWARE SWR IS NOT = -1
3263  005540  000403                           BR      66$             ;;BRANCH IF NO TIMEOUT
3264  005542  012716  005550           65$:    MOV     #66$,(SP)       ;;SET UP FOR TRAP RETURN
3265  005546  000002                           RTI
3266  005550  012737  000176  001154   66$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
3267  005556  012737  000174  001156           MOV     #DISPREG,DISPLAY
3268  005564  012637  000004           67$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
3269
3270  005570  005037  001230                   CLR     $PASS           ;;CLEAR PASS COUNT
3271  005574  132737  000200  001243           BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
3272  005602  001403                           BEQ     68$             ;;YES,USE NON-APT SWITCH
3273  005604  012737  001244  001154           MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
3274  005612                            68$:
```

```
3275                                    .SBTTL  TYPE PROGRAM NAME
3276                                    ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
3277  005612  005227  177777                    INC     #-1             ;;FIRST TIME?
3278  005616  001061                            BNE     69$             ;;BRANCH IF NO
3279  005620  022737  033156  000042            CMP     #$ENDAD,@#42    ;;ACT-11?
3280  005626  001455                            BEQ     69$             ;;BRANCH IF YES
3281  005630  104401  005676                    TYPE    ,70$            ;;TYPE ASCIZ STRING
3282                                    .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
3283  005634  005737  000042                    TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
3284  005640  001012                            BNE     71$             ;;BRANCH IF YES
3285  005642  123727  001242  000001            CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
3286  005650  001406                            BEQ     71$             ;;BRANCH IF YES
3287  005652  023727  001154  000176            CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
3288  005660  001005                            BNE     72$             ;;BRANCH IF NO
3289  005662  104407                            GTSWR                   ;;GET SOFT-SWR SETTINGS
3290  005664  000403                            BR      72$
3291  005666  112737  000001  001150  71$:     MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
3292  005674                          72$:
3293  005674  000432                            BR      69$             ;;GET OVER THE ASCIZ
3294                                  ;;70$:    .ASCIZ  <CRLF>@CZRMDB - RM03/RM02 SUBSYS FUNCTIONAL TEST,PART 2 @<CRLF>
3295  005762                          69$:
3296
3297
3298                                    ;FIND OUT IF PROGRAM IS RUNNING IN STANDALONE MODE
3299  005762  122737  000077  000041            CMPB    #77,@#41        ;LOAD FROM XXDP MEDIM ?
3300  005770  001003                            BNE     5$              ;BRANCH IF NOT
3301  005772  104401  064560                    TYPE    ,XDPMG          ;TYPE THE MESSAGE
3302  005776  000000                            HALT
3303  006000                          5$:
3304  006000  005737  000042                    TST     42              ;IS LOC 42 ZERO ??
3305  006004  001003                            BNE     10$             ;NO - NOT IN STANDALONE
3306  006006  105737  001242                    TSTB    $ENV            ;IS APT ENVIRONMENT ZERO ??
3307  006012  001454                            BEQ     STANDALONE      ;YES - PROGRAM IN STANDALONE
3308  006014                          10$:
3309
3310                                    ;PROGRAM NOT RUNNING IN STANDALONE - SEE IF SIZING IS ALLOWED
3311  006014  132737  000200  001243  XSIZ:     BITB    #BIT7,$ENVM     ;SIZING ALLOWED ??
3312  006022  001046                            BNE     20$             ;NO
3313  006024  012737  000377  001300            MOV     #377,$DEVM      ;YES - SET DEVICE MAP FOR ALL DEVICES
3314  006032  005037  001300                    CLR     $DEVM           ;CLEAR THE BIT MAP
3315  006036  005001                            CLR     R1              ;START FROM THE DRIVE 0
3316  006040  012704  000001                    MOV     #BIT0,R4        ;BIT MAP
3317  006044  013700  001276                    MOV     $BASE,R0        ;LOAD BASE ADDRESS
3318  006050  012760  000040  000010  15$:     MOV     #CLR,RMCS2(R0)  ;MASS BUS CLEAR
3319  006056  010160  000010                    MOV     R1,RMCS2(R0)    ;LOAD THE DRIVE ADDRESS
3320  006062  016003  000010                    MOV     RMCS2(R0),R3    ;READ DRIVE STATUS TO SEE NED BIT
3321  006066  032703  010000                    BIT     #NED,R3         ;NED BIT SET ?
3322  006072  001010                            BNE     16$             ;BRANCH IF SO
3323  006074  016003  000000                    MOV     RMCS1(R0),R3    ;CHECK THE DVA BIT
3324  006100  032703  004000                    BIT     #DVA,R3         ;DRIVE AVAILABLE ?
3325  006104  001403                            BEQ     16$             ;BRANCH IF NOT
3326  006106  050437  001300                    BIS     R4,$DEVM                ;SET THE BIT MAP
3327  006112  000405                            BR      17$             ;NOT TYPE THE NONE-EXIST MESG
3328  006114                          16$:
3329  006114  104401  064645                    TYPE    ,NOTEX          ;NOT EXIST DRIVE
3330  006120  010146                            MOV     R1,-(SP)        ;DRIVE NUMBER
```

```
 3331  006122  104403                    TYPOS
 3332  006124     006                    .BYTE   6
 3333  006125     000                    .BYTE   0
 3334  006126  005201              17$:   INC     R1              ;INCREMENT THE DRIVE ADDRESS
 3335  006130  006304                     ASL     R4              ;SET UP BIT MAP
 3336  006132  022701  000007             CMP     #7,R1           ;ALL DRIVE CHECKED ?
 3337  006136  103344                     BHIS    15$             ;BRANCH IF NOT
 3338  006140                       20$:
 3339
 3340                               ;GO TO COMMON START CODE
 3341  006140  000137  006772             JMP     CMNSTART
```

```
3342  006144                                  STANDALONE:
3343  006144  004737  061376                        JSR     PC,$TKINT       ;INITIALIZE CONSOLE
3344
3345                                          ;SEE IF THIS IS THE FIRST START AFTER PROGRAM WAS LOADED
3346  006150  005327  000001                        DEC     #1              ;FIRST START ??
3347  006154  100024                                BPL     10$             ;YES !!
3348
3349
3350                                          ;SEE IF THE USER WANTS TO KEEP SAME DEVICES FOR TESTING
3351  006156  104401  064067                 5$:    TYPE    ,CNSL00         ;MAINTAIN PREVIOUS PARAMETERS??
3352  006162  104411                                RDCHR                   ;GET RESPONSE
3353  006164  012637  001176                        MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3354  006170  104401  001176                        TYPE    ,$TMP1
3355  006174  123727  001176  000131                CMPB    $TMP1,#'Y       ;YES RESPONSE??
3356  006202  001002                                BNE     6$              ;NO!!
3357  006204  000137  007124                        JMP     READY           ;KEEP PREVIOUS PARAMETERS
3358  006210  123727  001176  000116          6$:   CMPB    $TMP1,#'N       ;NO RESPONSE??
3359  006216  001420                                BEQ     20$             ;GET NEW PARAMETERS
3360  006220  104401  063577                        TYPE    ,QSTMRK         ;NOT YES OR NO, TYPE "?"
3361  006224  000754                                BR      5$              ;RETRY
3362  006226                                  10$:
3363
3364                                          ;SEE IF OPERATOR WANTS HELP FILE
3365  006226  104401  063601                        TYPE    ,HELPQST        ;WANT HELP ??
3366  006232  104411                                RDCHR                   ;GET RESPONSE
3367  006234  012637  001176                        MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
3368  006240  104401  001176                        TYPE    ,$TMP1
3369  006244  123727  001176  000131                CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
3370  006252  001002                                BNE     20$             ;NO - DONT TYPE HELP
3371  006254  104401  103564                        TYPE    ,HELP           ;YES - TYPE HELP TEXT
3372  006260                                  20$:
3373
3374                                          ;SEE IF USER WANTS TO CHANGE RM03 UNIBUS ADDRESS
3375  006260  104401  063770                        TYPE    ,UBUSQST        ;WANT TO CHANGE ADDRESS ??
3376  006264  104411                                RDCHR                   ;GET RESPONSE
3377  006266  012637  001176                        MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
3378  006272  104401  001176                        TYPE    ,$TMP1
3379  006276  104411                                RDCHR                   ;READ ONE MORE CHARACTER
3380  006300  005726                                TST     (SP)+           ;CLEAR OFF THE STACK
3381  006302  123727  001176  000131                CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
3382  006310  001137                                BNE     90$             ;NO !!
3383  006312                                  30$:
3384
3385                                          ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, AND INTERRUPT VECTOR
3386  006312  104401  064126                        TYPE    ,CNSL01         ;TYPE CURRENT BUS ADDRESS
3387  006316  013746  001276                        MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
3388  006322  104402                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3389  006324  104401  063570                        TYPE    ,CLSPRN
3390  006330  104413                                RDOCT                   ;GET NEW BUS ADDRESS
3391  006332  012637  001176                        MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
3392  006336  001412                                BEQ     50$             ;YES-SKIP TO NEXT ENTRY
3393  006340  022737  160000  001176                CMP     #160000,$TMP1   ;BASE ADDRESS IN I/O PAGE??
3394  006346  101403                                BLOS    40$             ;YES
3395  006350  104401  064153                        TYPE    ,CNSL02         ;TYPE WARNING MESSAGE
3396  006354  000756                                BR      30$             ;RETRY
3397  006356  013737  001176  001276          40$:  MOV     $TMP1,$BASE     ;STORE NEW BUS ADDRESS
```

```
3398  006364  113737  001272  001176  50$:    MOVB    $VECT1,$TMP1    ;TYPE CURRENT VECTOR ADDRESS
3399  006372  105037  001177                  CLRB    $TMP1+1
3400  006376  104401  064234                  TYPE    ,CNSL03
3401  006402  013746  001176                  MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
3402  006406  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3403  006410     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
3404  006411     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3405  006412  104401  063570                  TYPE    ,CLSPRN
3406  006416  104413                          RDOCT                   ;GET NEW VECTOR ADDRESS
3407  006420  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN?
3408  006424  001412                          BEQ     70$             ;YES-SKIP TO NEXT ENTRY
3409  006426  022737  001000  001176          CMP     #1000,$TMP1     ;VECTOR ADDRESS < 1000??
3410  006434  101003                          BHI     60$             ;YES!!
3411  006436  104401  064264                  TYPE    ,CNSL04         ;TYPE WARNING MESSAGE
3412  006442  000750                          BR      50$             ;RETRY
3413  006444  113737  001176  001272  60$:    MOVB    $TMP1,$VECT1    ;STORE NEW VECTOR ADDRESS
3414  006452  113737  001273  001176  70$:    MOVB    $VECT1+1,$TMP1  ;TYPE CURRENT PRIORITY
3415  006460  006237  001176                  ASR     $TMP1
3416  006464  006237  001176                  ASR     $TMP1
3417  006470  006237  001176                  ASR     $TMP1
3418  006474  006237  001176                  ASR     $TMP1
3419  006500  006237  001176                  ASR     $TMP1
3420  006504  105037  001177                  CLRB    $TMP1+1
3421  006510  104401  064340                  TYPE    ,CNSL05
3422  006514  013746  001176                  MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
3423  006520  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3424  006522     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
3425  006523     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3426  006524  104401  063570                  TYPE    ,CLSPRN
3427  006530  104413                          RDOCT                   ;GET NEW PRIORITY
3428  006532  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
3429  006536  001424                          BEQ     90$             ;YES-SKIP TO NEXT ENTRY
3430  006540  023727  001176  000007          CMP     $TMP1,#7        ;LEGAL PRIORITY??
3431  006546  002403                          BLT     80$             ;YES!!
3432  006550  104401  064374                  TYPE    ,CNSL06         ;TYPE WARNING MESSAGE
3433  006554  000736                          BR      70$             ;RETRY
3434  006556                          80$:                            ;STORE NEW PRIORITY
3435  006556  006337  001176                  ASL     $TMP1
3436  006562  006337  001176                  ASL     $TMP1
3437  006566  006337  001176                  ASL     $TMP1
3438  006572  006337  001176                  ASL     $TMP1
3439  006576  006337  001176                  ASL     $TMP1
3440  006602  113737  001176  001273          MOVB    $TMP1,$VECT1+1
3441  006610                          90$:
3442
3443                                  ;DIALOGUE TO INPUT DEVICE NUMBERS
3444  006610  005037  001300                  CLR     $DEVM           ;CLEAR DEVICE MAP
3445  006614  104401  064421                  TYPE    ,CNSL07         ;TYPE INPUT INSTRUCTIONS
3446  006620  104401  063573                  TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
3447  006624  104411                          RDCHR                   ;GET RESPONSE
3448  006626  012637  001176                  MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3449  006632  104401  001176                  TYPE    ,$TMP1
3450  006636  023727  001176  000101          CMP     $TMP1,#'A       ;TEST ALL DRIVES??
3451  006644  001021                          BNE     110$            ;NO
3452  006646  000137  006014                  JMP     XSIZ            ;AUTO SIZE
3453  006652  012737  000377  001300          MOV     #377,$DEVM      ;TEST ALL DEVICES
```

```
3454  006660  000444                        BR      140$            ;SKIP TO NEXT ENTRY
3455  006662  104401  063573        100$:   TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
3456  006666  104411                        RDCHR                   ;GET RESPONSE
3457  006670  012637  001176                MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3458  006674  104401  001176                TYPE    ,$TMP1
3459  006700  023727  001176  000015        CMP     $TMP1,#CR       ;CARRIAGE RETURN??
3460  006706  001431                        BEQ     140$
3461  006710  023727  001176  000060  110$: CMP     $TMP1,#'0       ;NUMBER < 0??
3462  006716  002404                        BLT     120$            ;YES!!
3463  006720  023727  001176  000067        CMP     $TMP1,#'7       ;NUMBER > 7??
3464  006726  003403                        BLE     130$            ;NO!!
3465  006730  104401  063577        120$:   TYPE    ,QSTMRK         ;TYPE "?"
3466  006734  000752                        BR      100$            ;RETRY
3467  006736  013701  001176        130$:   MOV     $TMP1,R1        ;R1=DRIVE NUMBER
3468  006742  042701  177770                BIC     #^C7,R1
3469  006746  116102  064772                MOVB    ATNTBL(R1),R2           ;DECODE DEVICE NUMBER
3470  006752  042702  177400                BIC     #^C377,R2               ;CLEAR UNUSED BITS
3471  006756  050237  001300                BIS     R2,$DEVM                ;SET DEVICE # IN MAP
3472  006762  122737  000377  001300        CMPB    #377,$DEVM      ;DONE ??
3473  006770  101334                        BHI     100$            ;NO
3474  006772                        140$:
3475
```

```
3476  006772                                CMNSTART:
3477
3478                                        ;ASSEMBLE TEST QUE FROM DEVICE MAP
3479  006772  013700  001300                      MOV     $DEVM,R0          ;R0 = DEVICE MAP
3480  006776  012701  001452                      MOV     #TSTQUE+2,R1      ;R1 = ADDRESS OF FIRST ENTRY IN QUE
3481  007002  010137  001450                      MOV     R1,TSTQUE         ;INITIALIZE ENTRY POINTER
3482  007006  012702  000001                      MOV     #1,R2             ;R2 = DEVICE POINTER
3483  007012  005003                              CLR     R3                ;R3 = DEVICE NUMBER
3484  007014  030200                        10$:   BIT     R2,R0             ;IS THIS DEVICE IN MAP ??
3485  007016  001406                              BEQ     20$               ;NO !!
3486  007020  010311                              MOV     R3,(R1)           ;YES - ENTER DEVICE NUMBER IN QUE
3487  007022  116361  064772  000001              MOVB    ATNTBL(R3),1(R1);ENTER ATTENTION BIT IN QUE
3488  007030  062701  000002                      ADD     #2,R1             ;ADVANCE ENTRY POINTER
3489  007034  006302                        20$:   ASL     R2                ;ADVANCE DEVICE POINTER
3490  007036  105702                              TSTB    R2                ;DONE ALL DEVICES ??
3491  007040  001402                              BEQ     25$               ;YES
3492  007042  005203                              INC     R3                ;ADVANCE DEVICE NUMBER
3493  007044  000763                              BR      10$               ;ENTER NEXT DEVICE
3494  007046  005011                        25$:   CLR     (R1)              ;TERMINATE TEST QUE
3495
3496                                        ;SIZE FOR CLOCK
3497  007050  004737  040204                      JSR     PC,SIZCLK         ;SEE IF CLOCK PRESENT
3498  007054  000403                              BR      40$               ;YES - CLOCK IS PRESENT
3499  007056  104000                        30$:   ERROR                     ;NO CLOCK
3500  007060  000000                              HALT
3501  007062  000775                              BR      30$
3502  007064                                40$:
3503                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
3504  007064  005737  000042                      TST     @#42              ;;ARE WE RUNNING UNDER XXDP/ACT?
3505  007070  001012                              BNE     64$               ;;BRANCH IF YES
3506  007072  123727  001242  000001              CMPB    $ENV,#1           ;;ARE WE RUNNING UNDER APT?
3507  007100  001406                              BEQ     64$               ;;BRANCH IF YES
3508  007102  023727  001154  000176              CMP     SWR,#SWREG        ;;SOFTWARE SWITCH REG SELECTED?
3509  007110  001005                              BNE     65$               ;;BRANCH IF NO
3510  007112  104407                              GTSWR                     ;;GET SOFT-SWR SETTINGS
3511  007114  000403                              BR      65$
3512  007116  112737  000001  001150  64$:   MOVB    #1,$AUTOB         ;;SET AUTO-MODE INDICATOR
3513  007124                                65$:
3514  007124  000240                        READY: NOP                       ;READY TO START TEST
3515  007126  005037  001326                      CLR     CTLFG             ;CLEAR THE CONTROL-C FLAG
3516  007132  004737  061376                      JSR     PC,$TKINT         ;INITIALIZE TTY
3517  007136  013746  000300                      MOV     PR6,-(SP)         ;;PUT NEW PS ON STACK
3518  007142  012746  007150                      MOV     #64$,-(SP)        ;;PUT NEW PC ON  STACK
3519  007146  000002                              RTI                       ;;POP NEW PC AND PS
3520  007150                                64$:
3521  007150  117737  172274  001234              MOVB    @TSTQUE,$UNIT     ;LOAD UNIT NUMBER
3522  007156  005037  001474                      CLR     MEDENB            ;CLEAR MEDIA ENABLE
```

```
3523                                    ;;*****************************************************************
3524                                    ;*TEST 1        CONTROLLER ACCESS TEST
3525                                    ;;*****************************************************************
3526   007162                           TST1:
3527   007162  000240                           NOP                             ;START OF TEST
3528   007164  012737  007200  001122           MOV     #1$,$LPADR
3529   007172  012737  007200  001124           MOV     #1$,$LPERR
3530   007200                           1$:
3531   007200  012706  001100                   MOV     #STACK,SP               ;INITIALIZE STACK POINTER
3532   007204  013700  001276                   MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
3533   007210  013701  001450                   MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
3534   007214  012737  000001  001226           MOV     #1,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
3535   007222  005001                           CLR     R1
3536   007224  013746  000004                   MOV     ERRVEC,-(SP)            ;;PUSH ERRVEC ON STACK
3537   007230  013746  000006                   MOV     ERRVEC+2,-(SP)                  ;;PUSH ERRVEC+2 ON STACK
3538   007234  012737  007326  000004           MOV     #3$,ERRVEC
3539   007242  012737  000300  000006           MOV     #PR6,ERRVEC+2
3540
3541   007250  110160  000001                   MOVB    R1,RMCS1+1(R0)          ;MOVE HI BYTE TO RMCS1
3542   007254  010160  000002                   MOV     R1,RMWC(R0)             ;MOVE WORD COUNT REGISTER
3543   007260  016002  000002                   MOV     RMWC(R0),R2
3544   007264  010160  000004                   MOV     R1,RMBA(R0)             ;MOVE BUS ADDRESS REGISTER
3545   007270  016002  000004                   MOV     RMBA(R0),R2
3546   007274  010160  000010                   MOV     R1,RMCS2(R0)            ;MOVE CONTROL STATUS REGISTER
3547   007300  016002  000010                   MOV     RMCS2(R0),R2
3548   007304  010160  000022                   MOV     R1,RMDB(R0)             ;MOVE DATA BUFFER
3549   007310  016002  000022                   MOV     RMDB(R0),R2
3550   007314  012637  000006                   MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
3551   007320  012637  000004                   MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
3552   007324  000415                           BR      7$                      ;NO BUS TIMEOUT OCCURRED
3553
3554   007326  022626                   3$:     CMP     (SP)+,(SP)+             ;ADJUST STACK
3555   007330  012637  000006                   MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
3556   007334  012637  000004                   MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
3557   007340  104110                           ERROR   110                     ;CANNOT ACCESS MASSBUS CONTROLLER
3558   007342  005737  000042                   TST     42                      ;STAND ALONE MODE??
3559   007346  001002                           BNE     5$                      ;NO!!
3560   007350  000137  005324                   JMP     START                   ;YES-GO GET $BASE
3561   007354  000137  032766           5$:     JMP     $EOP                    ;GO TO END OF PASS HANDLER
3562   007360                           7$:
3563
3564                                    ;;*****************************************************************
3565                                    ;*TEST 2        DEVICE AVAILABLE TEST
3566
3567                                    ;;*****************************************************************
3568   007360                           TST2:
3569   007360  000004                           SCOPE                           ;SCOPE CALL
3570   007362  000240                           NOP                             ;START OF TEST
3571   007364  012706  001100                   MOV     #STACK,SP               ;INITIALIZE STACK POINTER
3572   007370  013700  001276                   MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
3573   007374  013701  001450                   MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
3574   007400  012737  000002  001226           MOV     #2,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
3575
3576   007406  004737  046660                   JSR     PC,CNTCLR
3577   007412  000404                           BR      2$      ;GO TO 2$ IF NO ERROR
3578   007414  000240                           NOP                             ;RETURN HERE IF ERROR
```

```
3579  007416  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
3580  007420  000137  007540                  JMP     7$      ;GO TO 7$ IF ERROR WAS FOUND
3581  007424                        2$:
3582  007424  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
3583  007430  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
3584  007434  012737  007524  000004          MOV     #5$,ERRVEC
3585  007442  013737  000300  000006          MOV     PR6,ERRVEC+2
3586
3587  007450  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET DVA STATUS
3588  007456  016037  000010  001174          MOV     RMCS2(R0),$TMP0 ;GET NED STATUS
3589  007464  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
3590  007470  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
3591  007474  032737  010000  001174          BIT     #NED,$TMP0      ;NONEXISTENT DEVICE??
3592  007502  001402                          BEQ     3$              ;NO!!
3593  007504  104111                          ERROR   111             ;NONEXISTENT DEVICE
3594  007506  000414                          BR      7$
3595  007510  032737  004000  001176  3$:     BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
3596  007516  001012                          BNE     9$              ;YES!!
3597  007520  104112                          ERROR   112             ;DEVICE NOT AVAILABLE
3598  007522  000406                          BR      7$
3599
3600  007524  022626                  5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
3601  007526  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
3602  007532  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
3603  007536  104113                          ERROR   113             ;BUS TIMEOUT (04 TRAP)
3604  007540  000137  032732          7$:     JMP     $EOSP
3605
3606  007544                          9$:
3607
3608                                  ;;*******************************************************************
3609                                  ;*TEST 3        DRIVE TYPE TEST
3610
3611                                  ;;*******************************************************************
3612  007544                          TST3:
3613  007544  000004                          SCOPE                   ;SCOPE CALL
3614  007546  000240                          NOP                     ;START OF TEST
3615  007550  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3616  007554  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3617  007560  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3618  007564  012737  000003  001226          MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3619
3620  007572  004737  046660                  JSR     PC,CNTCLR
3621  007576  000404                          BR      2$      ;GO TO 2$ IF NO ERROR
3622  007600  000240                          NOP                     ;RETURN HERE IF ERROR
3623  007602  104000                          ERROR                   ;ERROR NUMBER DEFINED BY SUB
3624  007604  000137  007722                  JMP     4$      ;GO TO 4$ IF ERROR WAS FOUND
3625  007610                          2$:
3626  007610  112737  000026  001506          MOVB    #RMDT,GETINX    ;SETUP GET INDEX TABLE
3627  007616  112737  000200  001507          MOVB    #200,GETINX+1
3628  007624  012737  007726  001356          MOV     #5$,RMDTI       ;RMDT INPUT BUFFER = 5$
3629  007632  004737  037516                  JSR     PC,GET          ;GO GET DRIVE TYPE
3630  007636  000402                          BR      3$      ;GO TO 3$ IF NO ERROR
3631  007640  000240                          NOP                     ;RETURN HERE IF ERROR
3632  007642  104000                          ERROR                   ;ERROR NUMBER SPECIFIED BY GET
3633  007644  022737  020024  001356  3$:     CMP     #SNGPRT,RMDTI   ;SINGLE PORT RM03??
3634  007652  001425                          BEQ     5$              ;YES!!
```

```
3635  007654  022737  024024  001356          CMP     #DULPRT,RMDTI   ;DUAL PORT RM03??
3636  007662  001421                          BEQ     5$              ;YES!!
3637  007664  022737  020025  001356          CMP     #SNGPRT!BIT0,RMDTI
3638  007672  001415                          BEQ     5$
3639  007674  022737  024025  001356          CMP     #DULPRT!BIT0,RMDTI
3640  007702  001411                          BEQ     5$
3641  007704  012737  020024  001176          MOV     #SNGPRT,$TMP1
3642  007712  012737  024024  001200          MOV     #DULPRT,$TMP2
3643  007720  104114                          ERROR   114             ;NOT AN RM03
3644  007722  000137  032732          4$:     JMP     $EOSP           ;GO TO SUBPASS HANDLER.
3645
3646  007726                          5$:
3647                                  ;;*****************************************************************
3648                                  ;*TEST 4        FORMAT ZEROS - 18
3649                                  ;;*****************************************************************
3650  007726                          TST4:
3651  007726  000004                          SCOPE                   ;SCOPE CALL
3652  007730  000240                          NOP                     ;START OF TEST
3653  007732  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3654  007736  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3655  007742  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3656  007746  012737  000004  001226          MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3657  007754                          10$:
3658
3659                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
3660  007754  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
3661  007762  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
3662  007770  012737  000000  001432          MOV     #0,RMOF0                ;18 BIT FORMAT
3663  007776  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
3664  010004  012737  103564  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
3665  010012  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
3666  010020  012737  065104  001174          MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
3667  010026  012737  000001  001176          MOV     #1,$TMP1
3668  010034  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
3669  010040                          20$:
3670
3671                                  ;PREPARE DEVICE FOR DATA TRANSFER
3672  010040  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
3673  010044  154130                          .WORD   154130  ;TASK DESCRIPTOR
3674  010046  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
3675  010050  000240                          NOP                     ;RETURN HERE IF ERROR
3676  010052  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3677  010054  000137  010526                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3678  010060                          30$:
3679
3680                                  ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
3681  010060  012737  000005  001400          MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
3682  010066  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
3683  010072  112722  000006                  MOVB    #RMDA,(R2)+
3684  010076  112722  000034                  MOVB    #RMDC,(R2)+
3685  010102  112722  000032                  MOVB    #RMOF,(R2)+
3686  010106  112722  000000                  MOVB    #RMCS1,(R2)+
3687  010112  112722  000200                  MOVB    #200,(R2)+
3688  010116  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
3689  010122  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
3690  010124  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
3691  010126  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
3692  010130  000137  010526                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3693  010134                          40$:
3694
3695                                  ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
3696  010134  004737  037432                  JSR     PC,GETSTS               ;SETUP FOR STATUS
3697  010140  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
3698  010144                          50$:
3699
3700                                  ;GO READ SEEK STATUS
3701  010144  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
3702  010150  000404                          BR      60$     ;GO TO 60$ IF NO ERROR
3703  010152  000240                          NOP                     ;RETURN HERE IF ERROR
3704  010154  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
3705  010156  000137  010526                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3706  010162                          60$:
3707
3708                                  ;VERIFY THE RESULTS OF THE SEEK COMMAND
3709  010162  004737  045420                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
3710  010166  000405                          BR      70$                     ;GO TO 70$ IF NO ERROR
3711  010170  000240                          NOP                     ;RETURN HERE IF ERROR
3712  010172  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
3713  010174  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3714  010176  000137  010526                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3715  010202                          70$:
3716
3717                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
3718  010202  012737  000063  001400          MOV     #WH!GO,RMCS1O           ;LOAD WRITE HEADER AND DATA
3719  010210  012702  001540                  MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
3720  010214  112722  000002                  MOVB    #RMWC,(R2)+
3721  010220  112722  000004                  MOVB    #RMBA,(R2)+
3722  010224  112722  000000                  MOVB    #RMCSI,(R2)+
3723  010230  112722  000200                  MOVB    #200,(R2)+
3724  010234  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
3725  010240  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
3726  010242  000240                          NOP                     ;RETURN HERE IF ERROR
3727  010244  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
3728  010246  000137  010526                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3729  010252                          80$:
3730
3731                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
3732  010252  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
3733  010256  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
3734  010262  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
3735  010264  000240                          NOP                     ;RETURN HERE IF ERROR
3736  010266  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
3737  010270  000137  010526                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3738  010274                          90$:
3739
3740                                  ;VERIFY RESULTS OF WRITE COMMAND
3741  010274  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
3742  010300  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
3743  010302  000240                          NOP                     ;RETURN HERE IF ERROR
3744  010304  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
3745  010306  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3746  010310  000137  010526                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
```

```
3747  010314                          100$:
3748  010314  004737  053016                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
3749  010320  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
3750  010322  000240                        NOP                     ;RETURN HERE IF ERROR
3751  010324  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3752  010326  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3753  010330  000137  010526                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3754  010334                          110$:
3755  010334  004737  041344                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
3756  010340  000405                        BR      120$                    ;GO TO 120$ IF NO ERROR
3757  010342  000240                        NOP                     ;RETURN HERE IF ERROR
3758  010344  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
3759  010346  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3760  010350  000137  010526                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3761  010354                          120$:
3762
3763
3764                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
3765  010354  012737  000073  001400        MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
3766  010362  012737  104570  001404        MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
3767  010370  004737  037766                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
3768  010374  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
3769  010376  000240                        NOP                     ;RETURN HERE IF ERROR
3770  010400  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
3771  010402  000137  010526                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3772  010406                          130$:
3773
3774                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
3775  010406  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
3776  010412  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
3777  010416  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
3778  010420  000240                        NOP                     ;RETURN HERE IF ERROR
3779  010422  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
3780  010424  000137  010526                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3781  010430                          140$:
3782
3783                                  ;VERIFY THE RESULTS OF READ OPERATION
3784  010430  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
3785  010434  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
3786  010436  000240                        NOP                     ;RETURN HERE IF ERROR
3787  010440  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
3788  010442  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3789  010444  000137  010526                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3790  010450                          150$:
3791  010450  004737  053016                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
3792  010454  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
3793  010456  000240                        NOP                     ;RETURN HERE IF ERROR
3794  010460  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3795  010462  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3796  010464  000137  010526                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3797  010470                          160$:
3798  010470  004737  041344                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
3799  010474  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
3800  010476  000240                        NOP                     ;RETURN HERE IF ERROR
3801  010500  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
3802  010502  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
3803  010504  000137  010526                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3804  010510                                170$:
3805
3806                                        ;VERIFY DATA
3807  010510  004737  037064                  JSR     PC,CMPBUF               ;GO COMPARE WRITE, READ DATA BUFFERS
3808  010514  103564                        .WORD   BUFONE                  ;STARTING ADDRESS OF WRITE BUFFER
3809  010516  104570                        .WORD   BUFTWO                  ;STARTING ADDDRESS OF READ BUFFER
3810  010520  000402                        BR      180$                            ;GO TO 180$ IF NO ERROR
3811  010522  000240                        NOP                             ;RETURN HERE IF ERROR
3812  010524  104000                        ERROR                           ;ERROR # DEFINED BY CMPBUF SUBROUTINE
3813  010526                                180$:
3814
3815                                        ;;*****************************************************************
3816                                        ;*TEST 5         FORMAT ZEROS - 16
3817                                        ;;*****************************************************************
3818  010526                                TST5:
3819  010526  000004                        SCOPE                           ;SCOPE CALL
3820  010530  000240                        NOP                             ;START OF TEST
3821  010532  012706  001100                MOV     #STACK,SP               ;INITIALIZE STACK POINTER
3822  010536  013700  001276                MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
3823  010542  013701  001450                MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
3824  010546  012737  000005  001226        MOV     #5,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
3825  010554                                10$:
3826
3827                                        ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
3828  010554  012737  000000  001434        MOV     #0,RMDC0                        ;CYLINDER = 0
3829  010562  012737  000000  001406        MOV     #0,RMDA0                        ;TRACK = SECTOR = 0
3830  010570  012737  010000  001432        MOV     #FMT16,RMOF0                    ;16 BIT FORMAT
3831  010576  012737  177376  001402        MOV     #<^C<2+256.>+1>,RMWC0           ;2 + 256 WORDS
3832  010604  012737  103564  001404        MOV     #BUFONE,RMBA0                   ;DATA BUFFER ADDRESS
3833  010612  012737  000062  001400        MOV     #WH,RMCS10                      ;WRITE HEADER AND DATA
3834  010620  012737  065104  001174        MOV     #ZEROS,$TMP0                    ;USE ALL ZEROS DATA PATTERN
3835  010626  012737  000001  001176        MOV     #1,$TMP1
3836  010634  004737  036620                MOV     PC,GENBUF                       ;GO GENERATE DATA BUFFER
3837  010640                                20$:
3838
3839                                        ;PREPARE DEVICE FOR DATA TRANSFER
3840  010640  004737  034000                JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
3841  010644  154130                        .WORD   154130  ;TASK DESCRIPTOR
3842  010646  000404                        BR      30$                             ;GO TO 30$ IF NO ERROR
3843  010650  000240                        NOP                             ;RETURN HERE IF ERROR
3844  010652  104000                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3845  010654  000137  011326                JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
3846  010660                                30$:
3847
3848                                        ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
3849  010660  012737  000005  001400        MOV     #SEEK!GO,RMCS10                 ;CHANGE COMMAND TO SEEK
3850  010666  012702  001535                MOV     #PUTINX,R2                      ;WRITE REGISTER INDEX TABLE
3851  010672  112722  000006                MOVB    #RMDA,(R2)+
3852  010676  112722  000034                MOVB    #RMDC,(R2)+
3853  010702  112722  000032                MOVB    #RMOF,(R2)+
3854  010706  112722  000000                MOVB    #RMCS1,(R2)+
3855  010712  112722  000200                MOVB    #200,(R2)+
3856  010716  004737  037766                JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
3857  010722  000404                        BR      40$     ;GO TO 40$ IF NO ERROR
3858  010724  000240                        NOP                             ;RETURN HERE IF ERROR
```

```
3859  010726  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
3860  010730  000137  011326                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3861  010734                          40$:
3862
3863                                  ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
3864  010734  004737  037432                  JSR     PC,GETSTS               ;SETUP FOR STATUS
3865  010740  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
3866  010744                          50$:
3867
3868                                  ;GO READ SEEK STATUS
3869  010744  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
3870  010750  000404                          BR      60$     ;GO TO 60$ IF NO ERROR
3871  010752  000240                          NOP                     ;RETURN HERE IF ERROR
3872  010754  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
3873  010756  000137  011326                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3874  010762                          60$:
3875
3876                                  ;VERIFY THE RESULTS OF THE SEEK COMMAND
3877  010762  004737  045420                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
3878  010766  000405                          BR      70$                     ;GO TO 70$ IF NO ERROR
3879  010770  000240                          NOP                     ;RETURN HERE IF ERROR
3880  010772  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
3881  010774  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3882  010776  000137  011326                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3883  011002                          70$:
3884
3885                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
3886  011002  012737  000063  001400          MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
3887  011010  012702  001540                  MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
3888  011014  112722  000002                  MOVB    #RMWC,(R2)+
3889  011020  112722  000004                  MOVB    #RMBA,(R2)+
3890  011024  112722  000000                  MOVB    #RMCS1,(R2)+
3891  011030  112722  000200                  MOVB    #200,(R2)+
3892  011034  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
3893  011040  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
3894  011042  000240                          NOP                     ;RETURN HERE IF ERROR
3895  011044  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
3896  011046  000137  011326                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3897  011052                          80$:
3898
3899                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
3900  011052  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
3901  011056  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
3902  011062  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
3903  011064  000240                          NOP                     ;RETURN HERE IF ERROR
3904  011066  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
3905  011070  000137  011326                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3906  011074                          90$:
3907
3908                                  ;VERIFY RESULTS OF WRITE COMMAND
3909  011074  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
3910  011100  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
3911  011102  000240                          NOP                     ;RETURN HERE IF ERROR
3912  011104  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
3913  011106  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3914  011110  000137  011326                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
```

```
3915  011114                          100$:
3916  011114  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
3917  011120  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
3918  011122  000240                          NOP                     ;RETURN HERE IF ERROR
3919  011124  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3920  011126  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3921  011130  000137  011326                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3922  011134                          110$:
3923  011134  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
3924  011140  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
3925  011142  000240                          NOP                     ;RETURN HERE IF ERROR
3926  011144  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
3927  011146  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3928  011150  000137  011326                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3929  011154                          120$:
3930
3931
3932                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
3933  011154  012737  000073  001400          MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
3934  011162  012737  104570  001404          MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
3935  011170  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
3936  011174  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
3937  011176  000240                          NOP                     ;RETURN HERE IF ERROR
3938  011200  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
3939  011202  000137  011326                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3940  011206                          130$:
3941
3942                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
3943  011206  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
3944  011212  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
3945  011216  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
3946  011220  000240                          NOP                     ;RETURN HERE IF ERROR
3947  011222  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
3948  011224  000137  011326                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3949  011230                          140$:
3950
3951                                  ;VERIFY THE RESULTS OF READ OPERATION
3952  011230  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
3953  011234  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
3954  011236  000240                          NOP                     ;RETURN HERE IF ERROR
3955  011240  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
3956  011242  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3957  011244  000137  011326                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3958  011250                          150$:
3959  011250  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
3960  011254  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
3961  011256  000240                          NOP                     ;RETURN HERE IF ERROR
3962  011260  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3963  011262  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3964  011264  000137  011326                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3965  011270                          160$:
3966  011270  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
3967  011274  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
3968  011276  000240                          NOP                     ;RETURN HERE IF ERROR
3969  011300  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
3970  011302  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
3971  011304  000137  011326          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3972  011310                        170$:
3973
3974                                ;VERIFY DATA
3975  011310  004737  037064          JSR     PC,CMPBUF               ;GO COMPARE WRITE, READ DATA BUFFERS
3976  011314  103564                  .WORD   BUFONE                  ;STARTING ADDRESS OF WRITE BUFFER
3977  011316  104570                  .WORD   BUFTWO                  ;STARTING ADDDRESS OF READ BUFFER
3978  011320  000402                  BR      180$                            ;GO TO 180$ IF NO ERROR
3979  011322  000240                  NOP                             ;RETURN HERE IF ERROR
3980  011324  104000                  ERROR                           ;ERROR # DEFINED BY CMPBUF SUBROUTINE
3981  011326                        180$:
3982
3983                                ;;*****************************************************************
3984                                ;*TEST 6         ZERO FILL TEST
3985                                ;;*****************************************************************
3986  011326                        TST6:
3987  011326  000004                  SCOPE                           ;SCOPE CALL
3988  011330  000240                  NOP                             ;START OF TEST
3989  011332  012706  001100          MOV     #STACK,SP               ;INITIALIZE STACK POINTER
3990  011336  013700  001276          MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
3991  011342  013701  001450          MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
3992  011346  012737  000006  001226  MOV     #6,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
3993  011354                        10$:
3994
3995                                ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
3996  011354  012737  000000  001434  MOV     #0,RMDC0                        ;CYLINDER = 0
3997  011362  012737  000000  001406  MOV     #0,RMDA0                        ;TRACK = SECTOR = 0
3998  011370  012737  010000  001432  MOV     #FMT16,RMOF0                    ;16 BIT FORMAT
3999  011376  012737  177376  001402  MOV     #<^C<2+256.>+1>,RMWC0           ;2 + 256 WORDS
4000  011404  012737  103564  001404  MOV     #BUFONE,RMBA0                   ;DATA BUFFER ADDRESS
4001  011412  012737  000062  001400  MOV     #WH,RMCS10                      ;WRITE HEADER AND DATA
4002  011420  012737  065104  001174  MOV     #ZEROS,$TMP0                    ;USE ALL ZEROS DATA PATTERN
4003  011426  012737  000001  001176  MOV     #1,$TMP1
4004  011434  004737  036620          JSR     PC,GENBUF                       ;GO GENERATE DATA BUFFER
4005  011440                        20$:
4006
4007                                ;PREPARE DEVICE FOR DATA TRANSFER
4008  011440  004737  034000          JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
4009  011444  154130                  .WORD   154130  ;TASK DESCRIPTOR
4010  011446  000404                  BR      30$                             ;GO TO 30$ IF NO ERROR
4011  011450  000240                  NOP                             ;RETURN HERE IF ERROR
4012  011452  104000                  ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4013  011454  000137  012142          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4014  011460                        30$:
4015
4016                                ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
4017  011460  012737  000005  001400  MOV     #SEEK!GO,RMCS10                 ;CHANGE COMMAND TO SEEK
4018  011466  012702  001535          MOV     #PUTINX,R2                      ;WRITE REGISTER INDEX TABLE
4019  011472  112722  000006          MOVB    #RMDA,(R2)+
4020  011476  112722  000034          MOVB    #RMDC,(R2)+
4021  011502  112722  000032          MOVB    #RMOF,(R2)+
4022  011506  112722  000000          MOVB    #RMCS1,(R2)+
4023  011512  112722  000200          MOVB    #200,(R2)+
4024  011516  004737  037766          JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
4025  011522  000404                  BR      40$     ;GO TO 40$ IF NO ERROR
4026  011524  000240                  NOP                             ;RETURN HERE IF ERROR
```

```
4027  011526  104000                        ERROR                           ;ERROR DEFINED BY PUT SUB
4028  011530  000137  012142                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4029  011534                          40$:
4030
4031                                  ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4032  011534  004737  037432                JSR     PC,GETSTS               ;SETUP FOR STATUS
4033  011540  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
4034  011544                          50$:
4035
4036                                  ;GO READ SEEK STATUS
4037  011544  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4038  011550  000404                        BR      60$     ;GO TO 60$ IF NO ERROR
4039  011552  000240                        NOP                     ;RETURN HERE IF ERROR
4040  011554  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
4041  011556  000137  012142                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4042  011562                          60$:
4043
4044                                  ;VERIFY THE RESULTS OF THE SEEK COMMAND
4045  011562  004737  045420                JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
4046  011566  000405                        BR      70$                     ;GO TO 70$ IF NO ERROR
4047  011570  000240                        NOP                     ;RETURN HERE IF ERROR
4048  011572  104000                        ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4049  011574  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4050  011576  000137  012142                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4051  011602                          70$:
4052
4053                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
4054  011602  012737  177776  001402        MOV     #<^C2+1>,RMWCO          ;FORMAT PARTIAL SECTOR
4055  011610  012737  000063  001400        MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
4056  011616  012702  001540                MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
4057  011622  112722  000002                MOVB    #RMWC,(R2)+
4058  011626  112722  000004                MOVB    #RMBA,(R2)+
4059  011632  112722  000000                MOVB    #RMCS1,(R2)+
4060  011636  112722  000200                MOVB    #200,(R2)+
4061  011642  004737  037766                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4062  011646  000404                        BR      80$     ;GO TO 80$ IF NO ERROR
4063  011650  000240                        NOP                     ;RETURN HERE IF ERROR
4064  011652  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
4065  011654  000137  012142                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4066  011660                          80$:
4067
4068                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4069  011660  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4070  011664  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4071  011670  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
4072  011672  000240                        NOP                     ;RETURN HERE IF ERROR
4073  011674  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
4074  011676  000137  012142                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4075  011702                          90$:
4076
4077                                  ;VERIFY RESULTS OF WRITE COMMAND
4078  011702  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4079  011706  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
4080  011710  000240                        NOP                     ;RETURN HERE IF ERROR
4081  011712  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4082  011714  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
4083  011716  000137  012142                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4084  011722                        100$:
4085  011722  004737  053016                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4086  011726  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
4087  011730  000240                        NOP                     ;RETURN HERE IF ERROR
4088  011732  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4089  011734  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4090  011736  000137  012142                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4091  011742                        110$:
4092  011742  004737  041344                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4093  011746  000405                        BR      120$                    ;GO TO 120$ IF NO ERROR
4094  011750  000240                        NOP                     ;RETURN HERE IF ERROR
4095  011752  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4096  011754  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4097  011756  000137  012142                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4098  011762                        120$:
4099
4100
4101                                ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
4102  011762  012737  177376  001402        MOV     #<^C<2+256.>+1>,RMWCO   ;READ A FULL SECTOR
4103  011770  012737  000073  001400        MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
4104  011776  012737  104570  001404        MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
4105  012004  004737  037766                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4106  012010  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
4107  012012  000240                        NOP                     ;RETURN HERE IF ERROR
4108  012014  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
4109  012016  000137  012142                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4110  012022                        130$:
4111
4112                                ;WAIT FOR READ TO COMPLETE AND GET STATUS
4113  012022  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
4114  012026  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4115  012032  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
4116  012034  000240                        NOP                     ;RETURN HERE IF ERROR
4117  012036  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
4118  012040  000137  012142                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4119  012044                        140$:
4120
4121                                ;VERIFY THE RESULTS OF READ OPERATION
4122  012044  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4123  012050  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
4124  012052  000240                        NOP                     ;RETURN HERE IF ERROR
4125  012054  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4126  012056  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4127  012060  000137  012142                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4128  012064                        150$:
4129  012064  004737  053016                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4130  012070  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
4131  012072  000240                        NOP                     ;RETURN HERE IF ERROR
4132  012074  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4133  012076  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4134  012100  000137  012142                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4135  012104                        160$:
4136  012104  004737  041344                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4137  012110  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
4138  012112  000240                        NOP                     ;RETURN HERE IF ERROR
```

```
4139  012114  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4140  012116  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4141  012120  000137  012142                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4142  012124                          170$:
4143
4144                                  ;VERIFY DATA
4145  012124  004737  037064                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
4146  012130  103564                          .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
4147  012132  104570                          .WORD   BUFTWO          ;STARTING ADDDRESS OF READ BUFFER
4148  012134  000402                          BR      180$                    ;GO TO 180$ IF NO ERROR
4149  012136  000240                          NOP                     ;RETURN HERE IF ERROR
4150  012140  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
4151  012142                          180$:
4152
4153                                  ;;**********************************************************************
4154                                  ;*TEST 7         FORMAT CHECK ZEROS - 16
4155                                  ;;**********************************************************************
4156  012142                          TST7:
4157  012142  000004                          SCOPE                   ;SCOPE CALL
4158  012144  000240                          NOP                     ;START OF TEST
4159  012146  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4160  012152  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4161  012156  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4162  012162  012737  000007  001226          MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4163  012170                          10$:
4164
4165                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
4166  012170  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
4167  012176  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
4168  012204  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
4169  012212  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
4170  012220  012737  103564  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
4171  012226  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
4172  012234  012737  065104  001174          MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
4173  012242  012737  000001  001176          MOV     #1,$TMP1
4174  012250  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
4175  012254                          20$:
4176
4177                                  ;PREPARE DEVICE FOR DATA TRANSFER
4178  012254  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4179  012260  154130                          .WORD   154130  ;TASK DESCRIPTOR
4180  012262  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
4181  012264  000240                          NOP                     ;RETURN HERE IF ERROR
4182  012266  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4183  012270  000137  012716                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4184  012274                          30$:
4185
4186                                  ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
4187  012274  012737  000005  001400          MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
4188  012302  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
4189  012306  112722  000006                  MOVB    #RMDA,(R2)+
4190  012312  112722  000034                  MOVB    #RMDC,(R2)+
4191  012316  112722  000032                  MOVB    #RMOF,(R2)+
4192  012322  112722  000000                  MOVB    #RMCS1,(R2)+
4193  012326  112722  000200                  MOVB    #200,(R2)+
4194  012332  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
```

```
4195  012336  000404                        BR      40$     ;GO TO 40$ IF NO ERROR
4196  012340  000240                        NOP                     ;RETURN HERE IF ERROR
4197  012342  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
4198  012344  000137  012716                JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4199  012350                        40$:
4200
4201                                ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4202  012350  004737  037432                JSR     PC,GETSTS               ;SETUP FOR STATUS
4203  012354  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
4204  012360                        50$:
4205
4206                                ;GO READ SEEK STATUS
4207  012360  004737  037516                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
4208  012364  000404                        BR      60$     ;GO TO 60$ IF NO ERROR
4209  012366  000240                        NOP                     ;RETURN HERE IF ERROR
4210  012370  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
4211  012372  000137  012716                JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4212  012376                        60$:
4213
4214                                ;VERIFY THE RESULTS OF THE SEEK COMMAND
4215  012376  004737  045420                JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
4216  012402  000405                        BR      70$                     ;GO TO 70$ IF NO ERROR
4217  012404  000240                        NOP                     ;RETURN HERE IF ERROR
4218  012406  104000                        ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4219  012410  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4220  012412  000137  012716                JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4221  012416                        70$:
4222
4223                                ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
4224  012416  012737  000063  001400        MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
4225  012424  012702  001540                MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
4226  012430  112722  000002                MOVB    #RMWC,(R2)+
4227  012434  112722  000004                MOVB    #RMBA,(R2)+
4228  012440  112722  000000                MOVB    #RMCS1,(R2)+
4229  012444  112722  000200                MOVB    #200,(R2)+
4230  012450  004737  037766                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4231  012454  000404                        BR      80$     ;GO TO 80$ IF NO ERROR
4232  012456  000240                        NOP                     ;RETURN HERE IF ERROR
4233  012460  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
4234  012462  000137  012716                JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4235  012466                        80$:
4236
4237                                ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4238  012466  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4239  012472  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4240  012476  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
4241  012500  000240                        NOP                     ;RETURN HERE IF ERROR
4242  012502  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
4243  012504  000137  012716                JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4244  012510                        90$:
4245
4246                                ;VERIFY RESULTS OF WRITE COMMAND
4247  012510  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4248  012514  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
4249  012516  000240                        NOP                     ;RETURN HERE IF ERROR
4250  012520  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
```

```
4251  012522  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4252  012524  000137  012716                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4253  012530                          100$:
4254  012530  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4255  012534  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
4256  012536  000240                          NOP                     ;RETURN HERE IF ERROR
4257  012540  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4258  012542  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4259  012544  000137  012716                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4260  012550                          110$:
4261  012550  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4262  012554  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
4263  012556  000240                          NOP                     ;RETURN HERE IF ERROR
4264  012560  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4265  012562  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4266  012564  000137  012716                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4267  012570                          120$:
4268
4269
4270                                  ;WRITE CHECK HEADER AND DATA FOR SECTOR JUST WRITTEN
4271  012570  012737  000053  001400          MOV     #WCH!GO,RMCS10          ;WRITE CHECK COMMAND
4272  012576  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4273  012602  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
4274  012604  000240                          NOP                     ;RETURN HERE IF ERROR
4275  012606  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4276  012610  000137  012716                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4277  012614                          130$:
4278
4279                                  ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
4280  012614  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
4281  012620  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4282  012624  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
4283  012626  000240                          NOP                     ;RETURN HERE IF ERROR
4284  012630  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4285  012632  000137  012716                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4286  012636                          140$:
4287
4288                                  ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
4289  012636  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4290  012642  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
4291  012644  000240                          NOP                     ;RETURN HERE IF ERROR
4292  012646  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4293  012650  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4294  012652  000137  012716                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4295  012656                          150$:
4296  012656  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4297  012662  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
4298  012664  000240                          NOP                     ;RETURN HERE IF ERROR
4299  012666  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4300  012670  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4301  012672  000137  012716                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4302  012676                          160$:
4303  012676  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4304  012702  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
4305  012704  000240                          NOP                     ;RETURN HERE IF ERROR
4306  012706  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
```

```
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
                                170$:

                                260$:

                                ;;*****************************************************************
                                ;*TEST 10        FORMAT CHECK ZEROS W/ WCE ERROR
                                ;;*****************************************************************
                                TST10:
                                        SCOPE                   ;SCOPE CALL
                                        NOP                     ;START OF TEST
                                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                                        MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                10$:

                                ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
                                        MOV     #0,RMDC0                ;CYLINDER = 0
                                        MOV     #0,RMDA0                ;TRACK = SECTOR = 0
                                        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
                                        MOV     #<^C<2*256.>+1>,RMWC0   ;2 * 256 WORDS
                                        MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
                                        MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
                                        MOV     #ZEROS,$TMP0            ;USE ALL ONES DATA PATTERN
                                        MOV     #1,$TMP1
                                        JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
                                20$:

                                ;PREPARE DEVICE FOR DATA TRANSFER
                                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                                        .WORD   154130  ;TASK DESCRIPTOR
                                        BR      30$                     ;GO TO 30$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
                                30$:

                                ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
                                        MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
                                        MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
                                        MOVB    #RMDA,(R2)+
                                        MOVB    #RMDC,(R2)+
                                        MOVB    #RMOF,(R2)+
                                        MOVB    #RMCS1,(R2)+
                                        MOVB    #200,(R2)+
                                        JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
                                        BR      40$     ;GO TO 40$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR DEFINED BY PUT SUB
                                        JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
                                40$:

                                ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
                                        JSR     PC,GETSTS               ;SETUP FOR STATUS
```

```
4363  013130  004737  040326          JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
4364  013134                      50$:
4365
4366                              ;GO READ SEEK STATUS
4367  013134  004737  037516          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4368  013140  000404                  BR      60$     ;GO TO 60$ IF NO ERROR
4369  013142  000240                  NOP                     ;RETURN HERE IF ERROR
4370  013144  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
4371  013146  000137  013636          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4372  013152                      60$:
4373
4374                              ;VERIFY THE RESULTS OF THE SEEK COMMAND
4375  013152  004737  045420          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
4376  013156  000405                  BR      70$                     ;GO TO 70$ IF NO ERROR
4377  013160  000240                  NOP                     ;RETURN HERE IF ERROR
4378  013162  104000                  ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4379  013164  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4380  013166  000137  013636          JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4381  013172                      70$:
4382
4383                              ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
4384  013172  012737  000063  001400      MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
```

```
4385  013200  012702  001540          MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
4386  013204  112722  000002          MOVB    #RMWC,(R2)+
4387  013210  112722  000004          MOVB    #RMBA,(R2)+
4388  013214  112722  000000          MOVB    #RMCS1,(R2)+
4389  013220  112722  000200          MOVB    #200,(R2)+
4390  013224  004737  037766          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4391  013230  000404                  BR      80$     ;GO TO 80$ IF NO ERROR
4392  013232  000240                  NOP                     ;RETURN HERE IF ERROR
4393  013234  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
4394  013236  000137  013636          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4395  013242                  80$:
4396
4397                          ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4398  013242  004737  040326          JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4399  013246  004737  037516          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4400  013252  000404                  BR      90$     ;GO TO 90$ IF NO ERROR
4401  013254  000240                  NOP                     ;RETURN HERE IF ERROR
4402  013256  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
4403  013260  000137  013636          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4404  013264                  90$:
4405
4406                          ;VERIFY RESULTS OF WRITE COMMAND
4407  013264  004737  040512          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4408  013270  000405                  BR      100$                    ;GO TO 100$ IF NO ERROR
4409  013272  000240                  NOP                     ;RETURN HERE IF ERROR
4410  013274  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4411  013276  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4412  013300  000137  013636          JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4413  013304                  100$:
4414  013304  004737  053016          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4415  013310  000405                  BR      110$                    ;GO TO 110$ IF NO ERROR
4416  013312  000240                  NOP                     ;RETURN HERE IF ERROR
4417  013314  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4418  013316  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4419  013320  000137  013636          JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4420  013324                  110$:
4421  013324  004737  041344          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4422  013330  000405                  BR      120$                    ;GO TO 120$ IF NO ERROR
4423  013332  000240                  NOP                     ;RETURN HERE IF ERROR
4424  013334  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4425  013336  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4426  013340  000137  013636          JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4427  013344                  120$:
4428
4429                          ;ALTER DATA BUFFER
4430  013344  005137  104566          COM     BUFTWO-2                ;COMPLEMENT LAST DATA WORD
4431
4432
4433                          ;SETUP AND WRITE CHECK HEADER AND DATA COMMAND
4434  013350  012737  000053  001400          MOV     #WCH!GO,RMCS10          ;LOAD COMMAND
4435  013356  004737  037766          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4436  013362  000404                  BR      180$    ;GO TO 180$ IF NO ERROR
4437  013364  000240                  NOP                     ;RETURN HERE IF ERROR
4438  013366  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
4439  013370  000137  013636          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4440  013374                  180$:
```

```
4441
4442                                    ;WAIT FOR WRITE CHECK COMMAND TO COMPLETE AND READ STATUS
4443  013374  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR COMMMAND TO COMPLETE
4444  013400  004737  037516                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4445  013404  000404                            BR      190$    ;GO TO 190$ IF NO ERROR
4446  013406  000240                            NOP                     ;RETURN HERE IF ERROR
4447  013410  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
4448  013412  000137  013636                    JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4449  013416                                190$:
4450
4451                                    ;CHECK FOR PRIMARY ERRORS
4452  013416  004737  040512                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4453  013422  000405                            BR      200$                    ;GO TO 200$ IF NO ERROR
4454  013424  000240                            NOP                     ;RETURN HERE IF ERROR
4455  013426  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4456  013430  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4457  013432  000137  013636                    JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4458  013436                                200$:
4459
```

```
4460                                    ;MAKE SURE THE WRITE CHECK ERROR WAS DETECTED
4461  013436  032737  040000  001340          BIT     #WCE,RMCS2I             ;IS WRITE CHECK ERROR SET??
4462  013444  001023                          BNE     210$                    ;YES!!
4463  013446  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4464  013452  000405                          BR      205$                    ;GO TO 205$ IF NO ERROR
4465  013454  000240                          NOP                     ;RETURN HERE IF ERROR
4466  013456  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4467  013460  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4468  013462  000137  013636                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4469  013466  013737  001340  001140  205$:   MOV     RMCS2I,$GDDAT           ;LOAD EXPECTED STATUS
4470  013474  052737  040000  001140          BIS     #WCE,$GDDAT
4471  013502  013737  001340  001142          MOV     RMCS2I,$BDDAT           ;LOAD RECEIVED STATUS
4472  013510  104337                          ERROR   337                     ;WRITE CHECK ERROR NOT DETECTED
4473  013512  000451                          BR      260$
4474  013514                          210$:
4475
4476                                    ;VERIFY THE ADDRESS OF THE WRITE CHECK ERROR
4477  013514  012737  104566  001134          MOV     #BUFTWO-2,$GDADR        ;LOAD EXPECTED ADDRESS
4478  013522  013737  001334  001136          MOV     RMBAI,$BDADR            ;LOAD RECEIVED ADDRESS
4479  013530  162737  000002  001136          SUB     #2,$BDADR               ;DECREMENT RECEIVED ADDRESS
4480
4481                                    ;GET WCE DATA AND VERIFY IT IS OK
4482  013536  112737  000022  001506          MOVB    #RMDB,GETINX            ;SETUP FOR READING RMDB
4483  013544  112737  000200  001507          MOVB    #200,GETINX+1
4484  013552  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4485  013556  000404                          BR      230$    ;GO TO 230$ IF NO ERROR
4486  013560  000240                          NOP                     ;RETURN HERE IF ERROR
4487  013562  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4488  013564  000137  013636                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4489  013570  013737  001352  001142  230$:   MOV     RMDB1,$BDDAT            ;LOAD RECEIVED DATA WORD
4490  013576  013737  104566  001140          MOV     BUFTWO-2,$GDDAT         ;LOAD EXPECTED DATA WORD
4491  013604  005137  001140                  COM     $GDDAT
4492  013610  023737  001134  001136          CMP     $GDADR,$BDADR           ;IS ADDRESS OK??
4493  013616  001402                          BEQ     220$                    ;YES!!
4494  013620  104340                          ERROR   340                     ;ADDRESS OF WCE INCORRECT
4495  013622  000405                          BR      260$
4496  013624  023737  001140  001142  220$:   CMP     $GDDAT,$BDDAT           ;IS DATA WORD OK??
4497  013632  001401                          BEQ     260$                    ;YES!!
4498  013634  104341                          ERROR   341                     ;UNEXPECTED WCE DATA
4499  013636                          260$:
4500
4501                                    ;;*******************************************************************
4502                                    ;*TEST 11        FORMAT ONES - 16
4503                                    ;;*******************************************************************
4504  013636                          TST11:
4505  013636  000004                          SCOPE                   ;SCOPE CALL
4506  013640  000240                          NOP                     ;START OF TEST
4507  013642  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4508  013646  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4509  013652  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4510  013656  012737  000011  001226          MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4511  013664                          10$:
4512
4513                                    ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
4514  013664  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
4515  013672  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
```

```
4516  013700  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
4517  013706  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
4518  013714  012737  103564  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
4519  013722  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
4520  013730  012737  065042  001174          MOV     #ONES,$TMP0             ;USE ALL ONES DATA PATTERN
4521  013736  012737  000001  001176          MOV     #1,$TMP1
4522  013744  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
4523  013750                                20$:
4524
4525                                        ;PREPARE DEVICE FOR DATA TRANSFER
4526  013750  004737  034000                      JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4527  013754  154130                              .WORD   154130  ;TASK DESCRIPTOR
4528  013756  000404                              BR      30$                     ;GO TO 30$ IF NO ERROR
4529  013760  000240                              NOP                     ;RETURN HERE IF ERROR
4530  013762  104000                              ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4531  013764  000137  014436                      JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4532  013770                                30$:
4533
4534                                        ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
4535  013770  012737  000005  001400          MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
4536  013776  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
4537  014002  112722  000006                  MOVB    #RMDA,(R2)+
4538  014006  112722  000034                  MOVB    #RMDC,(R2)+
4539  014012  112722  000032                  MOVB    #RMOF,(R2)+
4540  014016  112722  000000                  MOVB    #RMCS1,(R2)+
4541  014022  112722  000200                  MOVB    #200,(R2)+
4542  014026  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
4543  014032  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
4544  014034  000240                          NOP                             ;RETURN HERE IF ERROR
4545  014036  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
4546  014040  000137  014436                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4547  014044                                40$:
4548
4549                                        ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4550  014044  004737  037432                  JSR     PC,GETSTS               ;SETUP FOR STATUS
4551  014050  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
4552  014054                                50$:
4553
4554                                        ;GO READ SEEK STATUS
4555  014054  004737  037516                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
4556  014060  000404                          BR      60$     ;GO TO 60$ IF NO ERROR
4557  014062  000240                          NOP                             ;RETURN HERE IF ERROR
4558  014064  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
4559  014066  000137  014436                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4560  014072                                60$:
4561
4562                                        ;VERIFY THE RESULTS OF THE SEEK COMMAND
4563  014072  004737  045420                  JSR     PC,SEKSTS               ;GO VERIFY RESULTS OF SEEK OPERATION
4564  014076  000405                          BR      70$                             ;GO TO 70$ IF NO ERROR
4565  014100  000240                          NOP                             ;RETURN HERE IF ERROR
4566  014102  104000                          ERROR                           ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4567  014104  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4568  014106  000137  014436                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4569  014112                                70$:
4570
4571                                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
```

```
4572  014112  012737  000063  001400          MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
4573  014120  012702  001540                  MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
4574  014124  112722  000002                  MOVB    #RMWC,(R2)+
4575  014130  112722  000004                  MOVB    #RMBA,(R2)+
4576  014134  112722  000000                  MOVB    #RMCS1,(R2)+
4577  014140  112722  000200                  MOVB    #200,(R2)+
4578  014144  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4579  014150  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
4580  014152  000240                          NOP                     ;RETURN HERE IF ERROR
4581  014154  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4582  014156  000137  014436                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4583  014162                          80$:
4584
4585                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4586  014162  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4587  014166  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4588  014172  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
4589  014174  000240                          NOP                     ;RETURN HERE IF ERROR
4590  014176  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4591  014200  000137  014436                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4592  014204                          90$:
4593
4594                                  ;VERIFY RESULTS OF WRITE COMMAND
4595  014204  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4596  014210  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
4597  014212  000240                          NOP                     ;RETURN HERE IF ERROR
4598  014214  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4599  014216  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4600  014220  000137  014436                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4601  014224                          100$:
4602  014224  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4603  014230  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
4604  014232  000240                          NOP                     ;RETURN HERE IF ERROR
4605  014234  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4606  014236  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4607  014240  000137  014436                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4608  014244                          110$:
4609  014244  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4610  014250  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
4611  014252  000240                          NOP                     ;RETURN HERE IF ERROR
4612  014254  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4613  014256  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4614  014260  000137  014436                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4615  014264                          120$:
4616
4617
4618                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
4619  014264  012737  000073  001400          MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
4620  014272  012737  104570  001404          MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
4621  014300  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4622  014304  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
4623  014306  000240                          NOP                     ;RETURN HERE IF ERROR
4624  014310  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4625  014312  000137  014436                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4626  014316                          130$:
4627
```

```
4628
4629  014316  004737  040326          ;WAIT FOR READ TO COMPLETE AND GET STATUS
4630  014322  004737  037516                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
4631  014326  000404                          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4632  014330  000240                          BR      140$    ;GO TO 140$ IF NO ERROR
4633  014332  104000                          NOP                     ;RETURN HERE IF ERROR
4634  014334  000137  014436                  ERROR                   ;ERROR DEFINED BY GET SUB
4635  014340                                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4636                                  140$:
4637
4638  014340  004737  040512          ;VERIFY THE RESULTS OF READ OPERATION
4639  014344  000405                          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4640  014346  000240                          BR      150$                    ;GO TO 150$ IF NO ERROR
4641  014350  104000                          NOP                     ;RETURN HERE IF ERROR
4642  014352  004736                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4643  014354  000137  014436                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4644  014360                                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4645  014360  004737  053016          150$:
4646  014364  000405                          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4647  014366  000240                          BR      160$                    ;GO TO 160$ IF NO ERROR
4648  014370  104000                          NOP                     ;RETURN HERE IF ERROR
4649  014372  004736                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4650  014374  000137  014436                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4651  014400                                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4652  014400  004737  041344          160$:
4653  014404  000405                          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4654  014406  000240                          BR      170$                    ;GO TO 170$ IF NO ERROR
4655  014410  104000                          NOP                     ;RETURN HERE IF ERROR
4656  014412  004736                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4657  014414  000137  014436                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4658  014420                                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4659                                  170$:
4660
4661  014420  004737  037064          ;VERIFY DATA
4662  014424  103564                          JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
4663  014426  104570                          .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
4664  014430  000402                          .WORD   BUFTWO          ;STARTING ADDDRESS OF READ BUFFER
4665  014432  000240                          BR      180$                    ;GO TO 180$ IF NO ERROR
4666  014434  104000                          NOP                     ;RETURN HERE IF ERROR
4667  014436                                  ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
4668                                  180$:
4669
4670                                  ;;*****************************************************************
4671                                  ;*TEST 12        FORMAT CHECK ONES - 16
4672  014436                          ;;*****************************************************************
4673  014436  000004                  TST12:
4674  014440  000240                          SCOPE                   ;SCOPE CALL
4675  014442  012706  001100                  NOP                     ;START OF TEST
4676  014446  013700  001276                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4677  014452  013701  001450                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4678  014456  012737  000012  001226          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4679  014464                                  MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4680                                  10$:
4681
4682  014464  012737  000000  001434  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
4683  014472  012737  000000  001406          MOV     #0,RMDC0                ;CYLINDER = 0
                                        MOV     #0,RMDA0                ;TRACK = SECTOR = 0
```

```
4684  014500  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
4685  014506  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
4686  014514  012737  103564  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
4687  014522  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
4688  014530  012737  065042  001174          MOV     #ONES,$TMP0             ;USE ALL ONES DATA PATTERN
4689  014536  012737  000001  001176          MOV     #1,$TMP1
4690  014544  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
4691  014550                                20$:
4692
4693                                        ;PREPARE DEVICE FOR DATA TRANSFER
4694  014550  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4695  014554  154130                          .WORD   154130  ;TASK DESCRIPTOR
4696  014556  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
4697  014560  000240                          NOP                     ;RETURN HERE IF ERROR
4698  014562  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4699  014564  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4700  014570                                30$:
4701
4702                                        ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
4703  014570  012737  000005  001400          MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
4704  014576  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
4705  014602  112722  000006                  MOVB    #RMDA,(R2)+
4706  014606  112722  000034                  MOVB    #RMDC,(R2)+
4707  014612  112722  000032                  MOVB    #RMOF,(R2)+
4708  014616  112722  000000                  MOVB    #RMCS1,(R2)+
4709  014622  112722  000200                  MOVB    #200,(R2)+
4710  014626  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4711  014632  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
4712  014634  000240                          NOP                     ;RETURN HERE IF ERROR
4713  014636  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4714  014640  000137  015212                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4715  014644                                40$:
4716
4717                                        ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4718  014644  004737  037432                  JSR     PC,GETSTS               ;SETUP FOR STATUS
4719  014650  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
4720  014654                                50$:
4721
4722                                        ;GO READ SEEK STATUS
4723  014654  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4724  014660  000404                          BR      60$     ;GO TO 60$ IF NO ERROR
4725  014662  000240                          NOP                     ;RETURN HERE IF ERROR
4726  014664  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4727  014666  000137  015212                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4728  014672                                60$:
4729
4730                                        ;VERIFY THE RESULTS OF THE SEEK COMMAND
4731  014672  004737  045420                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
4732  014676  000405                          BR      70$                     ;GO TO 70$ IF NO ERROR
4733  014700  000240                          NOP                     ;RETURN HERE IF ERROR
4734  014702  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4735  014704  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4736  014706  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4737  014712                                70$:
4738
4739                                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
```

```
4740  014712  012737  000063  001400          MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
4741  014720  012702  001540                  MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
4742  014724  112722  000002                  MOVB    #RMWC,(R2)+
4743  014730  112722  000004                  MOVB    #RMBA,(R2)+
4744  014734  112722  000000                  MOVB    #RMCS1,(R2)+
4745  014740  112722  000200                  MOVB    #200,(R2)+
4746  014744  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4747  014750  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
4748  014752  000240                          NOP                     ;RETURN HERE IF ERROR
4749  014754  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4750  014756  000137  015212                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4751  014762                          80$:
4752
4753                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4754  014762  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4755  014766  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4756  014772  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
4757  014774  000240                          NOP                     ;RETURN HERE IF ERROR
4758  014776  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4759  015000  000137  015212                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4760  015004                          90$:
4761
4762                                  ;VERIFY RESULTS OF WRITE COMMAND
4763  015004  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4764  015010  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
4765  015012  000240                          NOP                     ;RETURN HERE IF ERROR
4766  015014  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4767  015016  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4768  015020  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4769  015024                          100$:
4770  015024  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4771  015030  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
4772  015032  000240                          NOP                     ;RETURN HERE IF ERROR
4773  015034  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4774  015036  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4775  015040  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4776  015044                          110$:
4777  015044  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4778  015050  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
4779  015052  000240                          NOP                     ;RETURN HERE IF ERROR
4780  015054  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4781  015056  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4782  015060  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4783  015064                          120$:
4784
4785
4786                                  ;WRITE CHECK HEADER AND DATA FOR SECTOR JUST WRITTEN
4787  015064  012737  000053  001400          MOV     #WCH!GO,RMCS10          ;WRITE CHECK COMMAND
4788  015072  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4789  015076  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
4790  015100  000240                          NOP                     ;RETURN HERE IF ERROR
4791  015102  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4792  015104  000137  015212                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4793  015110                          130$:
4794
4795                                  ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
```

```
4796  015110  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
4797  015114  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4798  015120  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
4799  015122  000240                          NOP                             ;RETURN HERE IF ERROR
4800  015124  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
4801  015126  000137  015212                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4802  015132                          140$:
4803
4804                                  ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
4805  015132  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4806  015136  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
4807  015140  000240                          NOP                             ;RETURN HERE IF ERROR
4808  015142  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
4809  015144  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4810  015146  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4811  015152                          150$:
4812  015152  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4813  015156  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
4814  015160  000240                          NOP                             ;RETURN HERE IF ERROR
4815  015162  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
4816  015164  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4817  015166  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4818  015172                          160$:
4819  015172  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4820  015176  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
4821  015200  000240                          NOP                             ;RETURN HERE IF ERROR
4822  015202  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
4823  015204  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4824  015206  000137  015212                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4825  015212                          170$:
4826
4827  015212                          260$:
4828
4829                                  ;;*****************************************************************
4830                                  ;*TEST 13       FORMAT CHECK ONES W/ WCE ERRORS
4831                                  ;;*****************************************************************
4832  015212                          TST13:
4833  015212  000004                          SCOPE                           ;SCOPE CALL
4834  015214  000240                          NOP                             ;START OF TEST
4835  015216  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4836  015222  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4837  015226  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4838  015232  012737  000013  001226          MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4839  015240                          10$:
4840
4841                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
4842  015240  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
4843  015246  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
4844  015254  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
4845  015262  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
4846  015270  012737  103564  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
4847  015276  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
4848  015304  012737  065042  001174          MOV     #ONES,$TMP0             ;USE ALL ONES DATA PATTERN
4849  015312  012737  000001  001176          MOV     #1,$TMP1
4850  015320  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
4851  015324                          20$:
```

```
4852
4853                                    ;PREPARE DEVICE FOR DATA TRANSFER
4854  015324  004737  034000                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4855  015330  154130                            .WORD   154130  ;TASK DESCRIPTOR
4856  015332  000404                            BR      30$                     ;GO TO 30$ IF NO ERROR
4857  015334  000240                            NOP                     ;RETURN HERE IF ERROR
4858  015336  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4859  015340  000137  016130                    JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4860  015344                                30$:
4861
4862                                    ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
4863  015344  012737  000005  001400            MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
4864  015352  012702  001535                    MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
4865  015356  112722  000006                    MOVB    #RMDA,(R2)+
4866  015362  112722  000034                    MOVB    #RMDC,(R2)+
4867  015366  112722  000032                    MOVB    #RMOF,(R2)+
4868  015372  112722  000000                    MOVB    #RMCS1,(R2)+
4869  015376  112722  000200                    MOVB    #200,(R2)+
4870  015402  004737  037766                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4871  015406  000404                            BR      40$     ;GO TO 40$ IF NO ERROR
4872  015410  000240                            NOP                     ;RETURN HERE IF ERROR
4873  015412  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
4874  015414  000137  016130                    JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4875  015420                                40$:
4876
4877                                    ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4878  015420  004737  037432                    JSR     PC,GETSTS               ;SETUP FOR STATUS
4879  015424  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
4880  015430                                50$:
4881
4882                                    ;GO READ SEEK STATUS
4883  015430  004737  037516                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4884  015434  000404                            BR      60$     ;GO TO 60$ IF NO ERROR
4885  015436  000240                            NOP                     ;RETURN HERE IF ERROR
4886  015440  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
4887  015442  000137  016130                    JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4888  015446                                60$:
4889
4890                                    ;VERIFY THE RESULTS OF THE SEEK COMMAND
4891  015446  004737  045420                    JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
4892  015452  000405                            BR      70$                     ;GO TO 70$ IF NO ERROR
4893  015454  000240                            NOP                     ;RETURN HERE IF ERROR
4894  015456  104000                            ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4895  015460  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4896  015462  000137  016130                    JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4897  015466                                70$:
4898
4899                                    ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
4900  015466  012737  000063  001400            MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
4901  015474  012702  001540                    MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
4902  015500  112722  000002                    MOVB    #RMWC,(R2)+
4903  015504  112722  000004                    MOVB    #RMBA,(R2)+
4904  015510  112722  000000                    MOVB    #RMCS1,(R2)+
4905  015514  112722  000200                    MOVB    #200,(R2)+
4906  015520  004737  037766                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4907  015524  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
```

```
4908  015526  000240                          NOP                     ;RETURN HERE IF ERROR
4909  015530  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4910  015532  000137  016130                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4911  015536                          80$:
4912
4913                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4914  015536  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4915  015542  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4916  015546  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
4917  015550  000240                          NOP                     ;RETURN HERE IF ERROR
4918  015552  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4919  015554  000137  016130                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4920  015560                          90$:
4921
4922                                  ;VERIFY RESULTS OF WRITE COMMAND
4923  015560  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4924  015564  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
4925  015566  000240                          NOP                     ;RETURN HERE IF ERROR
4926  015570  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4927  015572  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4928  015574  000137  016130                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4929  015600                          100$:
4930  015600  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4931  015604  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
4932  015606  000240                          NOP                     ;RETURN HERE IF ERROR
4933  015610  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4934  015612  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4935  015614  000137  016130                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4936  015620                          110$:
4937  015620  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4938  015624  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
4939  015626  000240                          NOP                     ;RETURN HERE IF ERROR
4940  015630  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4941  015632  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4942  015634  000137  016130                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4943  015640                          120$:
4944
4945                                  ;ALTER DATA BUFFER
4946  015640  005137  104566                  COM     BUFTWO-2                ;COMPLEMENT DATA WORD
4947
4948
4949                                  ;SETUP AND WRITE CHECK HEADER AND DATA COMMAND
4950  015644  012737  000053  001400          MOV     #WCH!GO,RMCS10          ;LOAD COMMAND
4951  015652  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4952  015656  000404                          BR      180$    ;GO TO 180$ IF NO ERROR
4953  015660  000240                          NOP                     ;RETURN HERE IF ERROR
4954  015662  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4955  015664  000137  016130                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4956  015670                          180$:
4957
4958                                  ;WAIT FOR WRITE CHECK COMMAND TO COMPLETE AND READ STATUS
4959  015670  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMMAND TO COMPLETE
4960  015674  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4961  015700  000404                          BR      190$    ;GO TO 190$ IF NO ERROR
4962  015702  000240                          NOP                     ;RETURN HERE IF ERROR
4963  015704  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
```

```
4964  015706  000137  016130                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4965  015712                          190$:
4966
4967                                  ;CHECK FOR PRIMARY ERRORS
4968  015712  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4969  015716  000405                          BR      200$                    ;GO TO 200$ IF NO ERROR
4970  015720  000240                          NOP                     ;RETURN HERE IF ERROR
4971  015722  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4972  015724  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4973  015726  000137  016130                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4974  015732                          200$:
4975
4976                                  ;MAKE SURE THE WRITE CHECK ERROR WAS DETECTED
4977  015732  032737  040000  001340          BIT     #WCE,RMCS21             ;IS WRITE CHECK ERROR SET??
4978  015740  001022                          BNE     210$                    ;YES!!
4979  015742  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4980  015746  000405                          BR      205$                    ;GO TO 205$ IF NO ERROR
4981  015750  000240                          NOP                     ;RETURN HERE IF ERROR
4982  015752  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4983  015754  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4984  015756  000137  016130                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4985  015762  013737  001340  001140  205$:   MOV     RMCS21,$GDDAT           ;LOAD EXPECTED STATUS
4986  015770  052737  040000  001140          BIS     #WCE,$GDDAT
4987  015776  013737  001340  001142          MOV     RMCS21,$BDDAT           ;LOAD RECEIVED STATUS
4988  016004  104337                          ERROR   337                     ;WRITE CHECK ERROR NOT DETECTED
4989  016006                          210$:
4990
4991                                  ;VERIFY THE ADDRESS OF THE WRITE CHECK ERROR
4992  016006  012737  104566  001134          MOV     #BUFTWO-2,$GDADR        ;LOAD EXPECTED ADDRESS
4993  016014  013737  001334  001136          MOV     RMBAI,$BDADR            ;LOAD RECEIVED ADDRESS
4994  016022  162737  000002  001136          SUB     #2,$BDADR               ;DECREMENT RECEIVED ADDRESS
4995
4996                                  ;GET WCE DATA AND VERIFY IT IS OK
4997  016030  112737  000022  001506          MOVB    #RMDB,GETINX            ;SETUP FOR READING RMDB
4998  016036  112737  000200  001507          MOVB    #200,GETINX+1
4999  016044  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5000  016050  000404                          BR      230$    ;GO TO 230$ IF NO ERROR
5001  016052  000240                          NOP                     ;RETURN HERE IF ERROR
5002  016054  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5003  016056  000137  016130                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
5004  016062  013737  001352  001142  230$:   MOV     RMDBI,$BDDAT            ;LOAD RECEIVED DATA WORD
5005  016070  013737  104566  001140          MOV     BUFTWO-2,$GDDAT         ;LOAD EXPECTED DATA WORD
5006  016076  005137  001140                  COM     $GDDAT
5007  016102  023737  001134  001136          CMP     $GDADR,$BDADR           ;IS ADDRESS OK??
5008  016110  001402                          BEQ     220$                    ;YES!!
5009  016112  104340                          ERROR   340                     ;ADDRESS OF WCE INCORRECT
5010  016114  000405                          BR      260$
5011  016116                          220$:
5012  016116  023737  001140  001142          CMP     $GDDAT,$BDDAT           ;IS DATA WORD OK??
5013  016124  001401                          BEQ     260$                    ;YES!!
5014  016126  104341                          ERROR   341                     ;UNEXPECTED WCE DATA
5015  016130                          260$:
5016
5017                                  ;;*******************************************************************
5018                                  ;*TEST 14        FORMAT MULTIPLE SECTORS
5019                                  ;;*******************************************************************
```

```
5020  016130                                  TST14:
5021  016130  000004                                  SCOPE                   ;SCOPE CALL
5022  016132  000240                                  NOP                     ;START OF TEST
5023  016134  012706  001100                          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5024  016140  013700  001276                          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5025  016144  013701  001450                          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5026  016150  012737  000014  001226                  MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5027  016156                                  10$:
5028
5029                                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5030  016156  012737  000000  001434                  MOV     #0,RMDC0                ;CYLINDER = 0
5031  016164  012737  000000  001406                  MOV     #0,RMDA0                ;TRACK = SECTOR = 0
5032  016172  012737  010000  001432                  MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5033  016200  012737  176774  001402                  MOV     #<^C<2+256.>*2>+1>,RMWC0;WORD COUNT FOR 2 SECTORS
5034  016206  012737  103564  001404                  MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
5035  016214  012737  000062  001400                  MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
5036  016222  012737  065104  001174                  MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
5037  016230  012737  000001  001176                  MOV     #1,$TMP1
5038  016236  004737  036620                          JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5039  016242                                  20$:
5040
5041                                          ;PREPARE DEVICE FOR DATA TRANSFER
5042  016242  004737  034000                          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5043  016246  154130                                  .WORD   154130  ;TASK DESCRIPTOR
5044  016250  000404                                  BR      30$                     ;GO TO 30$ IF NO ERROR
5045  016252  000240                                  NOP                     ;RETURN HERE IF ERROR
5046  016254  104000                                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5047  016256  000137  016704                          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5048  016262                                  30$:
5049
5050                                          ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
5051  016262  012737  000005  001400                  MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
5052  016270  012702  001535                          MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5053  016274  112722  000006                          MOVB    #RMDA,(R2)+
5054  016300  112722  000034                          MOVB    #RMDC,(R2)+
5055  016304  112722  000032                          MOVB    #RMOF,(R2)+
5056  016310  112722  000000                          MOVB    #RMCS1,(R2)+
5057  016314  112722  000200                          MOVB    #200,(R2)+
5058  016320  004737  037766                          JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5059  016324  000404                                  BR      40$     ;GO TO 40$ IF NO ERROR
5060  016326  000240                                  NOP                             ;RETURN HERE IF ERROR
5061  016330  104000                                  ERROR                           ;ERROR DEFINED BY PUT SUB
5062  016332  000137  016704                          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5063  016336                                  40$:
5064
5065                                          ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
5066  016336  004737  037432                          JSR     PC,GETSTS               ;SETUP FOR STATUS
5067  016342  004737  040326                          JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
5068  016346                                  50$:
5069
5070                                          ;GO READ SEEK STATUS
5071  016346  004737  037516                          JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5072  016352  000404                                  BR      60$     ;GO TO 60$ IF NO ERROR
5073  016354  000240                                  NOP                             ;RETURN HERE IF ERROR
5074  016356  104000                                  ERROR                           ;ERROR DEFINED BY GET SUB
5075  016360  000137  016704                          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
```

```
5076  016364                          60$:
5077
5078                                  ;VERIFY THE RESULTS OF THE SEEK COMMAND
5079  016364  004737  045420                JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
5080  016370  000405                        BR      70$                     ;GO TO 70$ IF NO ERROR
5081  016372  000240                        NOP                     ;RETURN HERE IF ERROR
5082  016374  104000                        ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5083  016376  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5084  016400  000137  016704                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5085  016404                          70$:
5086
5087                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5088  016404  012737  000063  001400        MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
5089  016412  012702  001540                MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
5090  016416  112722  000002                MOVB    #RMWC,(R2)+
5091  016422  112722  000004                MOVB    #RMBA,(R2)+
5092  016426  112722  000000                MOVB    #RMCS1,(R2)+
5093  016432  112722  000200                MOVB    #200,(R2)+
5094  016436  004737  037766                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5095  016442  000404                        BR      80$     ;GO TO 80$ IF NO ERROR
5096  016444  000240                        NOP                     ;RETURN HERE IF ERROR
5097  016446  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
5098  016450  000137  016704                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5099  016454                          80$:
5100
5101                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5102  016454  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
5103  016460  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5104  016464  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
5105  016466  000240                        NOP                     ;RETURN HERE IF ERROR
5106  016470  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5107  016472  000137  016704                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5108  016476                          90$:
5109
5110                                  ;VERIFY RESULTS OF WRITE COMMAND
5111  016476  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5112  016502  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
5113  016504  000240                        NOP                     ;RETURN HERE IF ERROR
5114  016506  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5115  016510  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5116  016512  000137  016704                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5117  016516                          100$:
5118  016516  004737  053016                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5119  016522  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
5120  016524  000240                        NOP                     ;RETURN HERE IF ERROR
5121  016526  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5122  016530  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5123  016532  000137  016704                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5124  016536                          110$:
5125  016536  004737  041344                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5126  016542  000405                        BR      120$                    ;GO TO 120$ IF NO ERROR
5127  016544  000240                        NOP                     ;RETURN HERE IF ERROR
5128  016546  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5129  016550  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5130  016552  000137  016704                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5131  016556                          120$:
```

```
5132
5133
5134                                    ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
5135  016556  012737  000053  001400            MOV     #WCH!GO,RMCS1O          ;READ HEADER & DATA COMMAND
5136  016564  004737  037766                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5137  016570  000404                            BR      130$    ;GO TO 130$ IF NO ERROR
5138  016572  000240                            NOP                     ;RETURN HERE IF ERROR
5139  016574  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
5140  016576  000137  016704                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5141  016602                                130$:
5142
5143                                    ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
5144  016602  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR WRITE CHECK TO FINISH
5145  016606  004737  037516                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5146  016612  000404                            BR      140$    ;GO TO 140$ IF NO ERROR
5147  016614  000240                            NOP                     ;RETURN HERE IF ERROR
5148  016616  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
5149  016620  000137  016704                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5150  016624                                140$:
5151
5152                                    ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
5153  016624  004737  040512                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5154  016630  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
5155  016632  000240                            NOP                     ;RETURN HERE IF ERROR
5156  016634  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5157  016636  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5158  016640  000137  016704                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5159  016644                                150$:
5160  016644  004737  053016                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5161  016650  000405                            BR      160$                    ;GO TO 160$ IF NO ERROR
5162  016652  000240                            NOP                     ;RETURN HERE IF ERROR
5163  016654  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5164  016656  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5165  016660  000137  016704                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5166  016664                                160$:
5167  016664  004737  041344                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5168  016670  000405                            BR      170$                    ;GO TO 170$ IF NO ERROR
5169  016672  000240                            NOP                     ;RETURN HERE IF ERROR
5170  016674  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5171  016676  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5172  016700  000137  016704                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5173  016704                                170$:
5174  016704                                180$:
5175
5176                                    ;;*****************************************************************
5177                                    ;*TEST 15        FORMAT W/ HEAD SWITCHING
5178                                    ;;*****************************************************************
5179  016704                            TST15:
5180  016704  000004                            SCOPE                   ;SCOPE CALL
5181  016706  000240                            NOP                     ;START OF TEST
5182  016710  012706  001100                    MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5183  016714  013700  001276                    MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5184  016720  013701  001450                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5185  016724  012737  000015  001226            MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5186  016732                                10$:
5187
```

```
5188                                    ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5189  016732  012737  000000  001434            MOV     #0,RMDC0                ;CYLINDER = 0
5190  016740  012737  000037  001406            MOV     #31.,RMDA0              ;START AT LAST SECTOR
5191  016746  012737  010000  001432            MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5192  016754  012737  176774  001402            MOV     #<^C<<2+256.>*2>+1>,RMWC0;WORD COUNT FOR 2 SECTORS
5193  016762  012737  103564  001404            MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
5194  016770  012737  000062  001400            MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
5195  016776  012737  065104  001174            MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
5196  017004  012737  000001  001176            MOV     #1,$TMP1
5197  017012  004737  036620                    JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5198  017016                                20$:
5199
5200                                    ;PREPARE DEVICE FOR DATA TRANSFER
5201  017016  004737  034000                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5202  017022  154130                            .WORD   154130  ;TASK DESCRIPTOR
5203  017024  000404                            BR      30$                     ;GO TO 30$ IF NO ERROR
5204  017026  000240                            NOP                     ;RETURN HERE IF ERROR
5205  017030  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5206  017032  000137  017460                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5207  017036                                30$:
5208
5209                                    ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
5210  017036  012737  000005  001400            MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
5211  017044  012702  001535                    MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5212  017050  112722  000006                    MOVB    #RMDA,(R2)+
5213  017054  112722  000034                    MOVB    #RMDC,(R2)+
5214  017060  112722  000032                    MOVB    #RMOF,(R2)+
5215  017064  112722  000000                    MOVB    #RMCS1,(R2)+
5216  017070  112722  000200                    MOVB    #200,(R2)+
5217  017074  004737  037766                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5218  017100  000404                            BR      40$     ;GO TO 40$ IF NO ERROR
5219  017102  000240                            NOP                     ;RETURN HERE IF ERROR
5220  017104  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
5221  017106  000137  017460                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5222  017112                                40$:
5223
5224                                    ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
5225  017112  004737  037432                    JSR     PC,GETSTS               ;SETUP FOR STATUS
5226  017116  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
5227  017122                                50$:
5228
5229                                    ;GO READ SEEK STATUS
5230  017122  004737  037516                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5231  017126  000404                            BR      60$     ;GO TO 60$ IF NO ERROR
5232  017130  000240                            NOP                     ;RETURN HERE IF ERROR
5233  017132  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
5234  017134  000137  017460                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5235  017140                                60$:
5236
5237                                    ;VERIFY THE RESULTS OF THE SEEK COMMAND
5238  017140  004737  045420                    JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
5239  017144  000405                            BR      70$                     ;GO TO 70$ IF NO ERROR
5240  017146  000240                            NOP                     ;RETURN HERE IF ERROR
5241  017150  104000                            ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5242  017152  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5243  017154  000137  017460                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
```

```
5244  017160                          70$:
5245
5246                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5247  017160  012737  000063  001400          MOV     #WH!GO,RMCS1O           ;LOAD WRITE HEADER AND DATA
5248  017166  012702  001540                  MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
5249  017172  112722  000002                  MOVB    #RMWC,(R2)+
5250  017176  112722  000004                  MOVB    #RMBA,(R2)+
5251  017202  112722  000000                  MOVB    #RMCS1,(R2)+
5252  017206  112722  000200                  MOVB    #200,(R2)+
5253  017212  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5254  017216  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
5255  017220  000240                          NOP                     ;RETURN HERE IF ERROR
5256  017222  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
5257  017224  000137  017460                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5258  017230                          80$:
5259
5260                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5261  017230  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
5262  017234  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5263  017240  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
5264  017242  000240                          NOP                     ;RETURN HERE IF ERROR
5265  017244  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5266  017246  000137  017460                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5267  017252                          90$:
5268
5269                                  ;VERIFY RESULTS OF WRITE COMMAND
5270  017252  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5271  017256  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
5272  017260  000240                          NOP                     ;RETURN HERE IF ERROR
5273  017262  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5274  017264  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5275  017266  000137  017460                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5276  017272                          100$:
5277  017272  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5278  017276  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
5279  017300  000240                          NOP                     ;RETURN HERE IF ERROR
5280  017302  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5281  017304  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5282  017306  000137  017460                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5283  017312                          110$:
5284  017312  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5285  017316  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
5286  017320  000240                          NOP                     ;RETURN HERE IF ERROR
5287  017322  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5288  017324  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5289  017326  000137  017460                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5290  017332                          120$:
5291
5292
5293                                  ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
5294  017332  012737  000053  001400          MOV     #WCH!GO,RMCS1O          ;READ HEADER & DATA COMMAND
5295  017340  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5296  017344  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
5297  017346  000240                          NOP                     ;RETURN HERE IF ERROR
5298  017350  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
5299  017352  000137  017460                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
```

```
5300  017356                          130$:
5301
5302                                  ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
5303  017356  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR WRITE CHECK TO FINISH
5304  017362  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5305  017366  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
5306  017370  000240                        NOP                     ;RETURN HERE IF ERROR
5307  017372  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5308  017374  000137  017460                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5309  017400                          140$:
5310
5311                                  ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
5312  017400  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5313  017404  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
5314  017406  000240                        NOP                     ;RETURN HERE IF ERROR
5315  017410  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5316  017412  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5317  017414  000137  017460                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5318  017420                          150$:
5319  017420  004737  053016                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5320  017424  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
5321  017426  000240                        NOP                     ;RETURN HERE IF ERROR
5322  017430  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5323  017432  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5324  017434  000137  017460                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5325  017440                          160$:
5326  017440  004737  041344                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5327  017444  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
5328  017446  000240                        NOP                     ;RETURN HERE IF ERROR
5329  017450  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5330  017452  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5331  017454  000137  017460                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5332  017460                          170$:
5333  017460                          180$:
5334
5335                                  ;;*****************************************************************
5336                                  ;*TEST 16        FORMAT W/ MID TRANSFER SEEK
5337                                  ;;*****************************************************************
5338  017460                          TST16:
5339  017460  000004                        SCOPE                   ;SCOPE CALL
5340  017462  000240                        NOP                     ;START OF TEST
5341  017464  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5342  017470  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5343  017474  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5344  017500  012737  000016  001226        MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5345  017506                          10$:
5346
5347                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5348  017506  012737  000000  001434        MOV     #0,RMDC0                ;CYLINDER = 0
5349  017514  012737  002037  001406        MOV     #002037,RMDA0           ;START AT LAST TRACK & SECTOR
5350  017522  012737  010000  001432        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5351  017530  012737  176774  001402        MOV     #<^C<<2+256.>*2>+1>,RMWC0;WORD COUNT FOR 2 SECTORS
5352  017536  012737  103564  001404        MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
5353  017544  012737  000062  001400        MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
5354  017552  012737  065104  001174        MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
5355  017560  012737  000001  001176        MOV     #1,$TMP1
```

```
5356  017566  004737  036620              JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5357  017572                          20$:
5358
5359                                  ;PREPARE DEVICE FOR DATA TRANSFER
5360  017572  004737  034000              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5361  017576  154130                      .WORD   154130  ;TASK DESCRIPTOR
5362  017600  000404                      BR      30$                     ;GO TO 30$ IF NO ERROR
5363  017602  000240                      NOP                     ;RETURN HERE IF ERROR
5364  017604  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5365  017606  000137  020234              JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5366  017612                          30$:
5367
5368                                  ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
5369  017612  012737  000005  001400      MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
5370  017620  012702  001535              MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5371  017624  112722  000006              MOVB    #RMDA,(R2)+
5372  017630  112722  000034              MOVB    #RMDC,(R2)+
5373  017634  112722  000032              MOVB    #RMOF,(R2)+
5374  017640  112722  000000              MOVB    #RMCS1,(R2)+
5375  017644  112722  000200              MOVB    #200,(R2)+
5376  017650  004737  037766              JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5377  017654  000404                      BR      40$     ;GO TO 40$ IF NO ERROR
5378  017656  000240                      NOP                     ;RETURN HERE IF ERROR
5379  017660  104000                      ERROR                   ;ERROR DEFINED BY PUT SUB
5380  017662  000137  020234              JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5381  017666                          40$:
5382
5383                                  ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
5384  017666  004737  037432              JSR     PC,GETSTS               ;SETUP FOR STATUS
5385  017672  004737  040326              JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
5386  017676                          50$:
5387
5388                                  ;GO READ SEEK STATUS
5389  017676  004737  037516              JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5390  017702  000404                      BR      60$     ;GO TO 60$ IF NO ERROR
5391  017704  000240                      NOP                     ;RETURN HERE IF ERROR
5392  017706  104000                      ERROR                   ;ERROR DEFINED BY GET SUB
5393  017710  000137  020234              JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5394  017714                          60$:
5395
5396                                  ;VERIFY THE RESULTS OF THE SEEK COMMAND
5397  017714  004737  045420              JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
5398  017720  000405                      BR      70$                     ;GO TO 70$ IF NO ERROR
5399  017722  000240                      NOP                     ;RETURN HERE IF ERROR
5400  017724  104000                      ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5401  017726  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5402  017730  000137  020234              JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5403  017734                          70$:
5404
5405                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5406  017734  012737  000063  001400      MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
5407  017742  012702  001540              MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
5408  017746  112722  000002              MOVB    #RMWC,(R2)+
5409  017752  112722  000004              MOVB    #RMBA,(R2)+
5410  017756  112722  000000              MOVB    #RMCS1,(R2)+
5411  017762  112722  000200              MOVB    #200,(R2)+
```

```
5412  017766  004737  037766          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5413  017772  000404                  BR      80$     ;GO TO 80$ IF NO ERROR
5414  017774  000240                  NOP                     ;RETURN HERE IF ERROR
5415  017776  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
5416  020000  000137  020234          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5417  020004                  80$:
5418
5419                          ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5420  020004  004737  040326          JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
5421  020010  004737  037516          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5422  020014  000404                  BR      90$     ;GO TO 90$ IF NO ERROR
5423  020016  000240                  NOP                     ;RETURN HERE IF ERROR
5424  020020  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
5425  020022  000137  020234          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5426  020026                  90$:
5427
5428                          ;VERIFY RESULTS OF WRITE COMMAND
5429  020026  004737  040512          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5430  020032  000405                  BR      100$                    ;GO TO 100$ IF NO ERROR
5431  020034  000240                  NOP                     ;RETURN HERE IF ERROR
5432  020036  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5433  020040  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5434  020042  000137  020234          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5435  020046                  100$:
5436  020046  004737  053016          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5437  020052  000405                  BR      110$                    ;GO TO 110$ IF NO ERROR
5438  020054  000240                  NOP                     ;RETURN HERE IF ERROR
5439  020056  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5440  020060  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5441  020062  000137  020234          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5442  020066                  110$:
5443  020066  004737  041344          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5444  020072  000405                  BR      120$                    ;GO TO 120$ IF NO ERROR
5445  020074  000240                  NOP                     ;RETURN HERE IF ERROR
5446  020076  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5447  020100  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5448  020102  000137  020234          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5449  020106                  120$:
5450
5451
5452                          ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
5453  020106  012737  000053  001400          MOV     #WCH!GO,RMCS1O          ;READ HEADER & DATA COMMAND
5454  020114  004737  037766          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5455  020120  000404                  BR      130$    ;GO TO 130$ IF NO ERROR
5456  020122  000240                  NOP                     ;RETURN HERE IF ERROR
5457  020124  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
5458  020126  000137  020234          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5459  020132                  130$:
5460
5461                          ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
5462  020132  004737  040326          JSR     PC,TIMOUT               ;WAIT FOR WRITE CHECK TO FINISH
5463  020136  004737  037516          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5464  020142  000404                  BR      140$    ;GO TO 140$ IF NO ERROR
5465  020144  000240                  NOP                     ;RETURN HERE IF ERROR
5466  020146  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
5467  020150  000137  020234          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
```

```
5468  020154                                  140$:
5469
5470                                          ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
5471  020154  004737  040512                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5472  020160  000405                                BR      150$                    ;GO TO 150$ IF NO ERROR
5473  020162  000240                                NOP                     ;RETURN HERE IF ERROR
5474  020164  104000                                ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5475  020166  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5476  020170  000137  020234                        JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5477  020174                                  150$:
5478  020174  004737  053016                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5479  020200  000405                                BR      160$                    ;GO TO 160$ IF NO ERROR
5480  020202  000240                                NOP                     ;RETURN HERE IF ERROR
5481  020204  104000                                ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5482  020206  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5483  020210  000137  020234                        JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5484  020214                                  160$:
5485  020214  004737  041344                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5486  020220  000405                                BR      170$                    ;GO TO 170$ IF NO ERROR
5487  020222  000240                                NOP                     ;RETURN HERE IF ERROR
5488  020224  104000                                ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5489  020226  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5490  020230  000137  020234                        JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5491  020234                                  170$:
5492  020234                                  180$:
5493
5494                                          ;;*****************************************************************
5495                                          ;*TEST 17       FORMAT W/ IMPLIED SEEK
5496                                          ;;*****************************************************************
5497  020234                                  TST17:
5498  020234  000004                                SCOPE                   ;SCOPE CALL
5499  020236  000240                                NOP                     ;START OF TEST
5500  020240  012706  001100                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5501  020244  013700  001276                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5502  020250  013701  001450                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5503  020254  012737  000017  001226                MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5504  020262                                  10$:
5505
5506                                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5507  020262  012737  000000  001434                MOV     #0,RMDC0                ;CYLINDER = 0
5508  020270  012737  000000  001406                MOV     #0,RMDA0                ;TRACK = SECTOR = 0
5509  020276  012737  010000  001432                MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5510  020304  012737  176774  001402                MOV     #<^C<<2+256.>*2>+1>,RMWC0;WORD COUNT FOR 2 SECTORS
5511  020312  012737  103564  001404                MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
5512  020320  012737  000062  001400                MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
5513  020326  012737  065104  001174                MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
5514  020334  012737  000001  001176                MOV     #1,$TMP1
5515  020342  004737  036620                        JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5516  020346                                  20$:
5517
5518                                          ;PREPARE DEVICE FOR DATA TRANSFER
5519  020346  004737  034000                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5520  020352  154130                                .WORD   154130  ;TASK DESCRIPTOR
5521  020354  000404                                BR      30$                     ;GO TO 30$ IF NO ERROR
5522  020356  000240                                NOP                     ;RETURN HERE IF ERROR
5523  020360  104000                                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
5524  020362  000137  021032                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5525  020366                                30$:
5526
5527                                        ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE OFF CYLINDER
5528  020366  013737  001434  021034            MOV     RMDC0,190$              ;SAVE CYLINDER ADDRESS
5529  020374  012737  001466  001434            MOV     #822.,RMDC0             ;SEEK TO LAST CYLINDER
5530  020402  012737  000005  001400            MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
5531  020410  012702  001535                    MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5532  020414  112722  000006                    MOVB    #RMDA,(R2)+
5533  020420  112722  000034                    MOVB    #RMDC,(R2)+
5534  020424  112722  000032                    MOVB    #RMOF,(R2)+
5535  020430  112722  000000                    MOVB    #RMCS1,(R2)+
5536  020434  112722  000200                    MOVB    #200,(R2)+
5537  020440  004737  037766                    JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5538  020444  000404                            BR      40$     ;GO TO 40$ IF NO ERROR
5539  020446  000240                            NOP                             ;RETURN HERE IF ERROR
5540  020450  104000                            ERROR                           ;ERROR DEFINED BY PUT SUB
5541  020452  000137  021032                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5542  020456                                40$:
5543
5544                                        ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
5545  020456  004737  037432                    JSR     PC,GETSTS               ;SETUP FOR STATUS
5546  020462  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
5547  020466                                50$:
5548
5549                                        ;GO READ SEEK STATUS
5550  020466  004737  037516                    JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5551  020472  000404                            BR      60$     ;GO TO 60$ IF NO ERROR
5552  020474  000240                            NOP                             ;RETURN HERE IF ERROR
5553  020476  104000                            ERROR                           ;ERROR DEFINED BY GET SUB
5554  020500  000137  021032                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5555  020504                                60$:
5556
5557                                        ;VERIFY THE RESULTS OF THE SEEK COMMAND
5558  020504  004737  045420                    JSR     PC,SEKSTS               ;GO VERIFY RESULTS OF SEEK OPERATION
5559  020510  000405                            BR      70$                             ;GO TO 70$ IF NO ERROR
5560  020512  000240                            NOP                             ;RETURN HERE IF ERROR
5561  020514  104000                            ERROR                           ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5562  020516  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5563  020520  000137  021032                    JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
5564  020524                                70$:
5565
5566                                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5567  020524  013737  021034  001434            MOV     190$,RMDC0              ;RESTORE DISK ADDRESS
5568  020532  012737  000063  001400            MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
5569  020540  012702  001540                    MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
5570  020544  112722  000002                    MOVB    #RMWC,(R2)+
5571  020550  112722  000004                    MOVB    #RMBA,(R2)+
5572  020554  112722  000000                    MOVB    #RMCS1,(R2)+
5573  020560  112722  000200                    MOVB    #200,(R2)+
5574  020564  004737  037766                    JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5575  020570  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
5576  020572  000240                            NOP                             ;RETURN HERE IF ERROR
5577  020574  104000                            ERROR                           ;ERROR DEFINED BY PUT SUB
5578  020576  000137  021032                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5579  020602                                80$:
```

```
5580
5581                                    ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5582  020602  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
5583  020606  004737  037516                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5584  020612  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
5585  020614  000240                            NOP                     ;RETURN HERE IF ERROR
5586  020616  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
5587  020620  000137  021032                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5588  020624                                90$:
5589
5590                                    ;VERIFY RESULTS OF WRITE COMMAND
5591  020624  004737  040512                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5592  020630  000405                            BR      100$                    ;GO TO 100$ IF NO ERROR
5593  020632  000240                            NOP                     ;RETURN HERE IF ERROR
5594  020634  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5595  020636  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5596  020640  000137  021032                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5597  020644                                100$:
5598  020644  004737  053016                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5599  020650  000405                            BR      110$                    ;GO TO 110$ IF NO ERROR
5600  020652  000240                            NOP                     ;RETURN HERE IF ERROR
5601  020654  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5602  020656  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5603  020660  000137  021032                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5604  020664                                110$:
5605  020664  004737  041344                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5606  020670  000405                            BR      120$                    ;GO TO 120$ IF NO ERROR
5607  020672  000240                            NOP                     ;RETURN HERE IF ERROR
5608  020674  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5609  020676  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5610  020700  000137  021032                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5611  020704                                120$:
5612
5613
5614                                    ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
5615  020704  012737  000053  001400            MOV     #WCH!GO,RMCS10          ;READ HEADER & DATA COMMAND
5616  020712  004737  037766                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5617  020716  000404                            BR      130$    ;GO TO 130$ IF NO ERROR
5618  020720  000240                            NOP                     ;RETURN HERE IF ERROR
5619  020722  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
5620  020724  000137  021032                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5621  020730                                130$:
5622
5623                                    ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
5624  020730  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR WRITE CHECK TO FINISH
5625  020734  004737  037516                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5626  020740  000404                            BR      140$    ;GO TO 140$ IF NO ERROR
5627  020742  000240                            NOP                     ;RETURN HERE IF ERROR
5628  020744  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
5629  020746  000137  021032                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5630  020752                                140$:
5631
5632                                    ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
5633  020752  004737  040512                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5634  020756  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
5635  020760  000240                            NOP                     ;RETURN HERE IF ERROR
```

```
5636  020762  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5637  020764  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5638  020766  000137  021032                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5639  020772                          150$:
5640  020772  004737  053016                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5641  020776  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
5642  021000  000240                        NOP                     ;RETURN HERE IF ERROR
5643  021002  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5644  021004  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5645  021006  000137  021032                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5646  021012                          160$:
5647  021012  004737  041344                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5648  021016  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
5649  021020  000240                        NOP                     ;RETURN HERE IF ERROR
5650  021022  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5651  021024  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5652  021026  000137  021032                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5653  021032                          170$:
5654  021032                          180$:
5655  021032  000401                        BR      200$
5656  021034  000000                  190$: .WORD                           ;TEMPORARY STORAGE
5657  021036                          200$:
5658
5659                                  ;;*******************************************************************
5660                                  ;*TEST 20       FORMAT EACH SECTOR ADDRESS
5661                                  ;;*******************************************************************
5662  021036                          TST20:
5663  021036  000004                        SCOPE                   ;SCOPE CALL
5664  021040  000240                        NOP                     ;START OF TEST
5665  021042  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5666  021046  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5667  021052  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5668  021056  012737  000020  001226        MOV     #20,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5669  021064                          10$:
5670
5671                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5672  021064  012737  000000  001434        MOV     #0,RMDC0                ;CYLINDER = 0
5673  021072  012737  000000  001406        MOV     #0,RMDA0                ;TRACK = SECTOR = 0
5674  021100  012737  010000  001432  15$:  MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5675  021106  012737  177376  001402        MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
5676  021114  012737  103564  001404        MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
5677  021122  012737  000062  001400        MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
5678  021130  012737  001406  001174        MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
5679  021136  012737  000001  001176        MOV     #1,$TMP1
5680  021144  004737  036620                JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5681  021150                          20$:
5682
5683                                  ;PREPARE DEVICE FOR DATA TRANSFER
5684  021150  004737  034000                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5685  021154  154130                        .WORD   154130  ;TASK DESCRIPTOR
5686  021156  000404                        BR      30$                     ;GO TO 30$ IF NO ERROR
5687  021160  000240                        NOP                     ;RETURN HERE IF ERROR
5688  021162  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5689  021164  000137  021562                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5690  021170                          30$:
5691
```

```
5692
5693  021170  012737  000063  001400          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5694  021176  012702  001535                  MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
5695  021202  112722  000006                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5696  021206  112722  000034                  MOVB    #RMDA,(R2)+
5697  021212  112722  000032                  MOVB    #RMDC,(R2)+
5698  021216  112722  000002                  MOVB    #RMOF,(R2)+
5699  021222  112722  000004                  MOVB    #RMWC,(R2)+
5700  021226  112722  000000                  MOVB    #RMBA,(R2)+
5701  021232  112722  000200                  MOVB    #RMCS1,(R2)+
5702  021236  004737  037766                  MOVB    #200,(R2)+
5703  021242  000404                          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5704  021244  000240                          BR      80$     ;GO TO 80$ IF NO ERROR
5705  021246  104000                          NOP                     ;RETURN HERE IF ERROR
5706  021250  000137  021562                  ERROR                   ;ERROR DEFINED BY PUT SUB
5707  021254                                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5708                                  80$:
5709
5710  021254  004737  037432                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
5711                                          JSR     PC,GETSTS
5712
5713  021260  004737  040326                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5714  021264  004737  037516                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
5715  021270  000404                          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5716  021272  000240                          BR      90$     ;GO TO 90$ IF NO ERROR
5717  021274  104000                          NOP                     ;RETURN HERE IF ERROR
5718  021276  000137  021562                  ERROR                   ;ERROR DEFINED BY GET SUB
5719  021302                                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5720                                  90$:
5721
5722  021302  004737  040512                  ;VERIFY RESULTS OF WRITE COMMAND
5723  021306  000405                          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5724  021310  000240                          BR      100$                    ;GO TO 100$ IF NO ERROR
5725  021312  104000                          NOP                     ;RETURN HERE IF ERROR
5726  021314  004736                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5727  021316  000137  021562                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5728  021322                                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5729  021322  004737  053016                  100$:
5730  021326  000405                          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5731  021330  000240                          BR      110$                    ;GO TO 110$ IF NO ERROR
5732  021332  104000                          NOP                     ;RETURN HERE IF ERROR
5733  021334  004736                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5734  021336  000137  021562                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5735  021342                                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5736  021342  004737  041344                  110$:
5737  021346  000405                          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5738  021350  000240                          BR      120$                    ;GO TO 120$ IF NO ERROR
5739  021352  104000                          NOP                     ;RETURN HERE IF ERROR
5740  021354  004736                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5741  021356  000137  021562                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5742  021362                                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5743                                  120$:
5744
5745                                          ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
5746  021362  012737  000073  001400          MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
5747  021370  012737  104570  001404          MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
```

```
5748  021376  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5749  021402  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
5750  021404  000240                          NOP                     ;RETURN HERE IF ERROR
5751  021406  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
5752  021410  000137  021562                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5753  021414                          130$:
5754
5755                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
5756  021414  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
5757  021420  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5758  021424  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
5759  021426  000240                          NOP                     ;RETURN HERE IF ERROR
5760  021430  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5761  021432  000137  021562                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5762  021436                          140$:
5763
5764                                  ;VERIFY THE RESULTS OF READ OPERATION
5765  021436  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5766  021442  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
5767  021444  000240                          NOP                     ;RETURN HERE IF ERROR
5768  021446  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5769  021450  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5770  021452  000137  021562                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5771  021456                          150$:
5772  021456  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5773  021462  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
5774  021464  000240                          NOP                     ;RETURN HERE IF ERROR
5775  021466  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5776  021470  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5777  021472  000137  021562                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5778  021476                          160$:
5779  021476  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5780  021502  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
5781  021504  000240                          NOP                     ;RETURN HERE IF ERROR
5782  021506  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5783  021510  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5784  021512  000137  021562                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5785  021516                          170$:
5786
5787                                  ;VERIFY DATA
5788  021516  004737  037064                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
5789  021522  103564                          .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
5790  021524  104570                          .WORD   BUFTWO          ;STARTING ADDDRESS OF READ BUFFER
5791  021526  000404                          BR      180$                    ;GO TO 180$ IF NO ERROR
5792  021530  000240                          NOP                     ;RETURN HERE IF ERROR
5793  021532  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
5794  021534  000137  021562                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5795  021540                          180$:
5796
5797                                  ;INCREMENT ADDRESS AND FORMAT NEXT SECTOR
5798  021540  062737  000001  001406          ADD     #1,RMDAO                ;ADVANCE SECTOR COUNT
5799  021546  122737  000037  001406          CMPB    #31.,RMDAO              ;DONE ALL SECTORS??
5800  021554  103402                          BLO     190$                    ;YES!!
5801  021556  000137  021100                  JMP     15$                     ;GO DO NEXT SECTOR
5802  021562                          190$:
5803
```

```
5804                                    ;;*****************************************************************
5805                                    ;*TEST 21       FORMAT EACH TRACK ADDRESS
5806                                    ;;*****************************************************************
5807  021562                            TST21:
5808  021562  000004                           SCOPE                   ;SCOPE CALL
5809  021564  000240                           NOP                     ;START OF TEST
5810  021566  012706  001100                   MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5811  021572  013700  001276                   MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5812  021576  013701  001450                   MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5813  021602  012737  000021  00 226           MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5814  021610                            10$:
5815
5816                                    ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5817  021610  012737  000000  001434           MOV     #0,RMDC0                ;CYLINDER = 0
5818  021616  012737  000000  001406           MOV     #0,RMDA0                ;TRACK = SECTOR = 0
5819  021624  012737  010000  001432   15$:    MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5820  021632  012737  177376  001402           MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
5821  021640  012737  103564  001404           MOV     #BUF0NE,RMBA0           ;DATA BUFFER ADDRESS
5822  021646  012737  000062  001400           MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
5823  021654  012737  001406  001174           MOV     #RMDA0,$TMP0            ;USE TRACK FOR DATA PATTERN
5824  021662  012737  000001  001176           MOV     #1,$TMP1
5825  021670  004737  036620                   JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5826  021674                            20$:
5827
5828                                    ;PREPARE DEVICE FOR DATA TRANSFER
5829  021674  004737  034000                   JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5830  021700  154130                           .WORD   154130  ;TASK DESCRIPTOR
5831  021702  000404                           BR      30$                     ;GO TO 30$ IF NO ERROR
5832  021704  000240                           NOP                     ;RETURN HERE IF ERROR
5833  021706  104000                           ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5834  021710  000137  022306                   JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5835  021714                            30$:
5836
5837                                    ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5838  021714  012737  000063  001400           MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
5839  021722  012702  001535                   MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5840  021726  112722  000006                   MOVB    #RMDA,(R2)+
5841  021732  112722  000034                   MOVB    #RMDC,(R2)+
5842  021736  112722  000032                   MOVB    #RMOF,(R2)+
5843  021742  112722  000002                   MOVB    #RMWC,(R2)+
5844  021746  112722  000004                   MOVB    #RMBA,(R2)+
5845  021752  112722  000000                   MOVB    #RMCS1,(R2)+
5846  021756  112722  000200                   MOVB    #200,(R2)+
5847  021762  004737  037766                   JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5848  021766  000404                           BR      80$     ;GO TO 80$ IF NO ERROR
5849  021770  000240                           NOP                     ;RETURN HERE IF ERROR
5850  021772  104000                           ERROR                   ;ERROR DEFINED BY PUT SUB
5851  021774  000137  022306                   JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5852  022000                            80$:
5853
5854                                    ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
5855  022000  004737  037432                   JSR     PC,GETSTS
5856
5857                                    ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5858  022004  004737  040326                   JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
5859  022010  004737  037516                   JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
```

```
5860  022014  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
5861  022016  000240                        NOP                     ;RETURN HERE IF ERROR
5862  022020  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5863  022022  000137  022306                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5864  022026                        90$:
5865
5866                                ;VERIFY RESULTS OF WRITE COMMAND
5867  022026  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5868  022032  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
5869  022034  000240                        NOP                     ;RETURN HERE IF ERROR
5870  022036  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5871  022040  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5872  022042  000137  022306                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5873  022046                        100$:
5874  022046  004737  053016                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5875  022052  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
5876  022054  000240                        NOP                     ;RETURN HERE IF ERROR
5877  022056  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5878  022060  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5879  022062  000137  022306                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5880  022066                        110$:
5881  022066  004737  041344                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5882  022072  000405                        BR      120$                    ;GO TO 120$ IF NO ERROR
5883  022074  000240                        NOP                     ;RETURN HERE IF ERROR
5884  022076  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5885  022100  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5886  022102  000137  022306                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5887  022106                        120$:
5888
5889
5890                                ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
5891  022106  012737  000073  001400        MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
5892  022114  012737  104570  001404        MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
5893  022122  004737  037766                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5894  022126  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
5895  022130  000240                        NOP                     ;RETURN HERE IF ERROR
5896  022132  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
5897  022134  000137  022306                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5898  022140                        130$:
5899
5900                                ;WAIT FOR READ TO COMPLETE AND GET STATUS
5901  022140  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
5902  022144  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5903  022150  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
5904  022152  000240                        NOP                     ;RETURN HERE IF ERROR
5905  022154  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5906  022156  000137  022306                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5907  022162                        140$:
5908
5909                                ;VERIFY THE RESULTS OF READ OPERATION
5910  022162  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5911  022166  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
5912  022170  000240                        NOP                     ;RETURN HERE IF ERROR
5913  022172  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5914  022174  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5915  022176  000137  022306                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
```

```
5916  022202
5917  022202  004737  053016          150$:
5918  022206  000405                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5919  022210  000240                        BR      160$                    ;GO TO 160$ IF NO ERROR
5920  022212  104000                        NOP                     ;RETURN HERE IF ERROR
5921  022214  004736                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5922  022216  000137  022306              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5923  022222                              JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5924  022222  004737  041344          160$:
5925  022226  000405                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5926  022230  000240                        BR      170$                    ;GO TO 170$ IF NO ERROR
5927  022232  104000                        NOP                     ;RETURN HERE IF ERROR
5928  022234  004736                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5929  022236  000137  022306              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5930  022242                              JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5931                                    170$:
5932
5933  022242  004737  037064          ;VERIFY DATA
5934  022246  103564                        JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
5935  022250  104570                        .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
5936  022252  000404                        .WORD   BUFTWO          ;STARTING ADDDRESS OF READ BUFFER
5937  022254  000240                        BR      180$                    ;GO TO 180$ IF NO ERROR
5938  022256  104000                        NOP                     ;RETURN HERE IF ERROR
5939  022260  000137  022306              ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
5940  022264                              JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5941                                    180$:
5942
5943  022264  062737  000400  001406  ;INCREMENT ADDRESS AND FORMAT NEXT TRACK
5944  022272  022737  002000  001406        ADD     #400,RMDAO              ;ADVANCE TRACK COUNT
5945  022300  103402                        CMP     #2000,RMDAO             ;DONE ALL TRACKS??
5946  022302  000137  021624                BLO     190$                    ;YES!!
5947  022306                              JMP     15$                     ;GO DO NEXT SECTOR
5948                                    190$:
5949
5950                                    ;;*****************************************************************
5951                                    ;*TEST 22        FORMAT PRIME CYLINDERS
5952  022306                            ;;*****************************************************************
5953  022306  000004                    TST22:
5954  022310  000240                          SCOPE                   ;SCOPE CALL
5955  022312  012706  001100                  NOP                     ;START OF TEST
5956  022316  013700  001276                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5957  022322  013701  001450                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5958  022326  012737  000022  001226          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5959  022334                                  MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5960                                    10$:
5961
5962  022334  012737  000001  001434  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5963  022342  012737  000000  001406        MOV     #1,RMDC0                ;CYLINDER = 0
5964  022350  012737  010000  001432  15$:   MOV     #0,RMDA0                ;TRACK = SECTOR = 0
5965  022356  012737  177376  001402        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5966  022364  012737  103564  001404        MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
5967  022372  012737  000062  001400        MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
5968  022400  012737  001434  001174        MOV     #WH,RMCS1O              ;WRITE HEADER AND DATA
5969  022406  012737  000001  001176        MOV     #RMDC0,$TMP0            ;USE CYLINDER FOR DATA PATTERN
5970  022414  004737  036620                MOV     #1,$TMP1
5971  022420                                JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
                                      20$:
```

```
5972
5973                                    ;PREPARE DEVICE FOR DATA TRANSFER
5974  022420  004737  034000                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5975  022424  154130                            .WORD   154130  ;TASK DESCRIPTOR
5976  022426  000404                            BR      30$                     ;GO TO 30$ IF NO ERROR
5977  022430  000240                            NOP                     ;RETURN HERE IF ERROR
5978  022432  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5979  022434  000137  023030                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5980  022440                                30$:
5981
5982                                    ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5983  022440  012737  000063  001400            MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
5984  022446  012702  001535                    MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5985  022452  112722  000006                    MOVB    #RMDA,(R2)+
5986  022456  112722  000034                    MOVB    #RMDC,(R2)+
5987  022462  112722  000032                    MOVB    #RMOF,(R2)+
5988  022466  112722  000002                    MOVB    #RMWC,(R2)+
5989  022472  112722  000004                    MOVB    #RMBA,(R2)+
5990  022476  112722  000000                    MOVB    #RMCS1,(R2)+
5991  022502  112722  000200                    MOVB    #200,(R2)+
5992  022506  004737  037766                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5993  022512  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
5994  022514  000240                            NOP                     ;RETURN HERE IF ERROR
5995  022516  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
5996  022520  000137  023030                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5997  022524                                80$:
5998
5999                                    ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6000  022524  004737  037432                    JSR     PC,GETSTS
6001
6002                                    ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6003  022530  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6004  022534  004737  037516                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6005  022540  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
6006  022542  000240                            NOP                     ;RETURN HERE IF ERROR
6007  022544  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
6008  022546  000137  023030                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6009  022552                                90$:
6010
6011                                    ;VERIFY RESULTS OF WRITE COMMAND
6012  022552  004737  040512                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6013  022556  000405                            BR      100$                    ;GO TO 100$ IF NO ERROR
6014  022560  000240                            NOP                     ;RETURN HERE IF ERROR
6015  022562  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6016  022564  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6017  022566  000137  023030                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6018  022572                                100$:
6019  022572  004737  053016                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6020  022576  000405                            BR      110$                    ;GO TO 110$ IF NO ERROR
6021  022600  000240                            NOP                     ;RETURN HERE IF ERROR
6022  022602  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6023  022604  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6024  022606  000137  023030                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6025  022612                                110$:
6026  022612  004737  041344                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6027  022616  000405                            BR      120$                    ;GO TO 120$ IF NO ERROR
```

```
6028  022620  000240                          NOP                     ;RETURN HERE IF ERROR
6029  022622  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBRCUTINE
6030  022624  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6031  022626  000137  023030                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6032  022632                          120$:
6033
6034
6035                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6036  022632  012737  000073  001400          MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
6037  022640  012737  104570  001404          MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
6038  022646  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6039  022652  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
6040  022654  000240                          NOP                     ;RETURN HERE IF ERROR
6041  022656  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6042  022660  000137  023030                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6043  022664                          130$:
6044
6045                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
6046  022664  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6047  022670  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6048  022674  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
6049  022676  000240                          NOP                     ;RETURN HERE IF ERROR
6050  022700  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
6051  022702  000137  023030                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6052  022706                          140$:
6053
6054                                  ;VERIFY THE RESULTS OF READ OPERATION
6055  022706  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6056  022712  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
6057  022714  000240                          NOP                     ;RETURN HERE IF ERROR
6058  022716  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6059  022720  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6060  022722  000137  023030                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6061  022726                          150$:
6062  022726  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6063  022732  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
6064  022734  000240                          NOP                     ;RETURN HERE IF ERROR
6065  022736  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6066  022740  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6067  022742  000137  023030                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6068  022746                          160$:
6069  022746  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6070  022752  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
6071  022754  000240                          NOP                     ;RETURN HERE IF ERROR
6072  022756  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6073  022760  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6074  022762  000137  023030                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6075  022766                          170$:
6076
6077                                  ;VERIFY DATA
6078  022766  004737  037064                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
6079  022772  103564                          .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
6080  022774  104570                          .WORD   BUFTWO          ;STARTING ADDDRESS OF READ BUFFER
6081  022776  000404                          BR      180$                    ;GO TO 180$ IF NO ERROR
6082  023000  000240                          NOP                     ;RETURN HERE IF ERROR
6083  023002  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
```

```
6084  023004  000137  023030                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6085  023010                              180$:
6086
6087                                      ;INCREMENT ADDRESS AND FORMAT NEXT PRIME CYLINDER
6088  023010  006337  001434                  ASL     RMDC0                   ;ADVANCE CYLINDER COUNT
6089  023014  022737  001466  001434          CMP     #822.,RMDC0             ;DONE ALL CYLINDERS??
6090  023022  103402                          BLO     190$                    ;YES!!
6091  023024  000137  022350                  JMP     15$                     ;GO DO NEXT CYLINDER
6092  023030                              190$:
6093
6094                                      ;;**********************************************************************
6095                                      ;*TEST 23        FORMAT LAST SECTOR
6096                                      ;;**********************************************************************
6097  023030                              TST23:
6098  023030  000004                          SCOPE                           ;SCOPE CALL
6099  023032  000240                          NOP                             ;START OF TEST
6100  023034  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
6101  023040  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
6102  023044  013701  001450                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
6103  023050  012737  000023  001226          MOV     #23,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
6104  023056                              10$:
6105
6106                                      ;PREPARE DEVICE FOR DATA TRANSFER
6107  023056  004737  034000                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
6108  023062  154130                          .WORD   154130  ;TASK DESCRIPTOR
6109  023064  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
6110  023066  000240                          NOP                             ;RETURN HERE IF ERROR
6111  023070  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6112  023072  000137  023326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6113  023076                              30$:
6114
6115                                      ;SETUP PARAMETERS FOR READING LAST SECTOR
6116  023076  012737  001466  001434          MOV     #822.,RMDC0             ;LAST CYLINDER
6117  023104  012737  002037  001406          MOV     #2037,RMDA0             ;TRACK = 4 SECTOR = 31.
6118  023112  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
6119  023120  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
6120  023126  012737  104570  001404          MOV     #BUFTWO,RMBA0           ;DATA BUFFER ADDRESS
6121  023134  012737  000073  001400          MOV     #RH!GO,RMCS10           ;WRITE HEADER AND DATA
6122
6123  023142  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
6124  023146  112722  000006                  MOVB    #RMDA,(R2)+
6125  023152  112722  000034                  MOVB    #RMDC,(R2)+
6126  023156  112722  000032                  MOVB    #RMOF,(R2)+
6127  023162  112722  000004                  MOVB    #RMBA,(R2)+
6128  023166  112722  000002                  MOVB    #RMWC,(R2)+
6129  023172  112722  000000                  MOVB    #RMCS1,(R2)+
6130  023176  112722  000200                  MOVB    #200,(R2)+
6131
6132
6133  023202                              120$:
6134
6135                                      ;READ HEADER AND DATA OF LAST SECTOR
6136  023202  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
6137  023206  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
6138  023210  000240                          NOP                             ;RETURN HERE IF ERROR
6139  023212  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
```

```
6140  023214  000137  023326                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6141  023220                          130$:
6142
6143                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6144  023220  004737  037432                  JSR     PC,GETSTS
6145                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
6146  023224  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6147  023230  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6148  023234  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
6149  023236  000240                          NOP                     ;RETURN HERE IF ERROR
6150  023240  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
6151  023242  000137  023326                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6152  023246                          140$:
6153
6154                                  ;VERIFY THE RESULTS OF READ OPERATION
6155  023246  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6156  023252  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
6157  023254  000240                          NOP                     ;RETURN HERE IF ERROR
6158  023256  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6159  023260  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6160  023262  000137  023326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6161  023266                          150$:
6162  023266  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6163  023272  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
6164  023274  000240                          NOP                     ;RETURN HERE IF ERROR
6165  023276  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6166  023300  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6167  023302  000137  023326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6168  023306                          160$:
6169  023306  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6170  023312  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
6171  023314  000240                          NOP                     ;RETURN HERE IF ERROR
6172  023316  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6173  023320  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6174  023322  000137  023326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6175  023326                          170$:
6176
6177  023326                          190$:
6178
6179                                  ;;*******************************************************************
6180                                  ;*TEST 24        FORMAT W/ AOE ERROR
6181                                  ;;*******************************************************************
6182  023326                          TST24:
6183  023326  000004                          SCOPE                   ;SCOPE CALL
6184  023330  000240                          NOP                     ;START OF TEST
6185  023332  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6186  023336  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6187  023342  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6188  023346  012737  000024  001226          MOV     #24,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6189
6190                                  ;PREPARE DEVICE FOR DATA TRANSFER
6191  023354  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6192  023360  154130                          .WORD   154130  ;TASK DESCRIPTOR
6193  023362  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
6194  023364  000240                          NOP                     ;RETURN HERE IF ERROR
6195  023366  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
6196  023370  000137  023624                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6197  023374                          30$:
6198
6199                                  ;SETUP PARAMETERS FOR READING 2 SECTORS STARTING WITH LAST SECTOR
6200  023374  012737  001466  001434          MOV     #822.,RMDC0             ;START AT LAST CYLINDER
6201  023402  012737  002037  001406          MOV     #2037,RMDA0             ;TRACK = 4 SECTOR = 31.
6202  023410  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
6203  023416  012737  176774  001402          MOV     #<^C<<2+256.>*2>+1>,RMWC0        ;FORMAT 2 SECTORS
6204  023424  012737  104570  001404          MOV     #BUFTW0,RMBA0           ;DATA BUFFER ADDRESS
6205  023432  012737  000073  001400          MOV     #RH!GO,RMCS10           ;WRITE HEADER AND DATA
6206  023440  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
6207  023444  112722  000006                  MOVB    #RMDA,(R2)+
6208  023450  112722  000034                  MOVB    #RMDC,(R2)+
6209  023454  112722  000032                  MOVB    #RMOF,(R2)+
6210  023460  112722  000002                  MOVB    #RMWC,(R2)+
6211  023464  112722  000004                  MOVB    #RMBA,(R2)+
6212  023470  112722  000000                  MOVB    #RMCS1,(R2)+
6213  023474  112722  000200                  MOVB    #200,(R2)+
6214  023500                          80$:
6215
6216  023500                          125$:
6217  023500  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
6218  023504  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
6219  023506  000240                          NOP                             ;RETURN HERE IF ERROR
6220  023510  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
6221  023512  000137  023624                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6222  023516                          130$:
6223
6224                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6225  023516  004737  037432                  JSR     PC,GETSTS
6226
6227                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
6228  023522  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6229  023526  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6230  023532  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
6231  023534  000240                          NOP                             ;RETURN HERE IF ERROR
6232  023536  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
6233  023540  000137  023624                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6234  023544                          140$:
6235
6236                                  ;VERIFY THE RESULTS OF READ OPERATION
6237  023544  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6238  023550  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
6239  023552  000240                          NOP                     ;RETURN HERE IF ERROR
6240  023554  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6241  023556  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6242  023560  000137  023624                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6243  023564                          150$:
6244  023564  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6245  023570  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
6246  023572  000240                          NOP                     ;RETURN HERE IF ERROR
6247  023574  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6248  023576  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6249  023600  000137  023624                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6250  023604                          160$:
6251  023604  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
```

```
6252  023610  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
6253  023612  000240                          NOP                     ;RETURN HERE IF ERROR
6254  023614  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6255  023616  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6256  023620  000137  023624                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6257  023624                          170$:
6258
6259  023624                          190$:
6260
6261                                  ;;*****************************************************************
6262                                  ;*TEST 25        FORMAT INVALID SECTOR ADDRESS
6263                                  ;;*****************************************************************
6264  023624                          TST25:
6265  023624  000004                          SCOPE                   ;SCOPE CALL
6266  023626  000240                          NOP                     ;START OF TEST
6267  023630  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6268  023634  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6269  023640  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6270  023644  012737  000025  001226          MOV     #25,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6271  023652                          10$:
6272
6273                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6274  023652  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
6275  023660  012737  000036  001406          MOV     #30.,RMDA0              ;TRACK = 0, SECTOR = 30
6276  023666  012737  000000  001432  15$:    MOV     #0,RMOF0                ;18 BIT FORMAT
6277  023674  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
6278  023702  012737  103564  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
6279  023710  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
6280  023716  012737  001406  001174          MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
6281  023724  012737  000001  001176          MOV     #1,$TMP1
6282  023732  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
6283  023736                          20$:
6284
6285                                  ;PREPARE DEVICE FOR DATA TRANSFER
6286  023736  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6287  023742  154130                          .WORD   154130  ;TASK DESCRIPTOR
6288  023744  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
6289  023746  000240                          NOP                     ;RETURN HERE IF ERROR
6290  023750  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6291  023752  000137  024352                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6292  023756                          30$:
6293
6294                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
6295  023756  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
6296  023764  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
6297  023770  112722  000006                  MOVB    #RMDA,(R2)+
6298  023774  112722  000034                  MOVB    #RMDC,(R2)+
6299  024000  112722  000032                  MOVB    #RMOF,(R2)+
6300  024004  112722  000002                  MOVB    #RMWC,(R2)+
6301  024010  112722  000004                  MOVB    #RMBA,(R2)+
6302  024014  112722  000000                  MOVB    #RMCS1,(R2)+
6303  024020  112722  000200                  MOVB    #200,(R2)+
6304  024024  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6305  024030  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
6306  024032  000240                          NOP                     ;RETURN HERE IF ERROR
6307  024034  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
```

```
6308  024036  000137  024352                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6309  024042                          80$:
6310
6311                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6312  024042  004737  037432                  JSR     PC,GETSTS
6313
6314                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6315  024046  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6316  024052  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6317  024056  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
6318  024060  000240                          NOP                     ;RETURN HERE IF ERROR
6319  024062  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
6320  024064  000137  024352                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6321  024070                          90$:
6322
6323                                  ;VERIFY RESULTS OF WRITE COMMAND
6324  024070  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6325  024074  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
6326  024076  000240                          NOP                     ;RETURN HERE IF ERROR
6327  024100  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6328  024102  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6329  024104  000137  024352                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6330  024110                          100$:
6331  024110  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6332  024114  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
6333  024116  000240                          NOP                     ;RETURN HERE IF ERROR
6334  024120  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6335  024122  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6336  024124  000137  024352                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6337  024130                          110$:
6338  024130  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6339  024134  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
6340  024136  000240                          NOP                     ;RETURN HERE IF ERROR
6341  024140  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6342  024142  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6343  024144  000137  024352                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6344  024150                          120$:
6345
6346                                  ;CLEAR "IAE" ERROR AND PREPARE DEVICE
6347  024150  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6348  024154  154130                          .WORD   154130  ;TASK DESCRIPTOR
6349  024156  000404                          BR      125$                    ;GO TO 125$ IF NO ERROR
6350  024160  000240                          NOP                     ;RETURN HERE IF ERROR
6351  024162  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6352  024164  000137  024352                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6353  024170                          125$:
6354
6355                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6356  024170  012737  000073  001400          MOV     #RH!GO,RMCS10                   ;READ HEADER & DATA COMMAND
6357  024176  012737  104570  001404          MOV     #BUFTWO,RMBA0                   ;CHANGE BUS ADDRESS
6358  024204  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6359  024210  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
6360  024212  000240                          NOP                     ;RETURN HERE IF ERROR
6361  024214  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6362  024216  000137  024352                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6363  024222                          130$:
```

```
6364
6365                                    ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6366  024222  004737  037432                    JSR     PC,GETSTS
6367
6368                                    ;WAIT FOR READ TO COMPLETE AND GET STATUS
6369  024226  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6370  024232  004737  037516                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6371  024236  000404                            BR      140$    ;GO TO 140$ IF NO ERROR
6372  024240  000240                            NOP                     ;RETURN HERE IF ERROR
6373  024242  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
6374  024244  000137  024352                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6375  024250                                140$:
6376
6377                                    ;VERIFY THE RESULTS OF READ OPERATION
6378  024250  004737  040512                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6379  024254  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
6380  024256  000240                            NOP                     ;RETURN HERE IF ERROR
6381  024260  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6382  024262  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6383  024264  000137  024352                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6384  024270                                150$:
6385  024270  004737  053016                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6386  024274  000405                            BR      160$                    ;GO TO 160$ IF NO ERROR
6387  024276  000240                            NOP                     ;RETURN HERE IF ERROR
6388  024300  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6389  024302  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6390  024304  000137  024352                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6391  024310                                160$:
6392  024310  004737  041344                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6393  024314  000405                            BR      170$                    ;GO TO 170$ IF NO ERROR
6394  024316  000240                            NOP                     ;RETURN HERE IF ERROR
6395  024320  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6396  024322  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6397  024324  000137  024352                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6398  024330                                170$:
6399  024330                                180$:
6400
6401                                    ;INCREMENT ADDRESS AND FORMAT NEXT SECTOR
6402  024330  062737  000001  001406            ADD     #1,RMDA0                ;ADVANCE SECTOR COUNT
6403  024336  022737  000037  001406            CMP     #31.,RMDA0              ;DONE ALL SECTORS??
6404  024344  103402                            BLO     190$                    ;YES!!
6405  024346  000137  023666                    JMP     15$                     ;GO DO NEXT SECTOR
6406  024352                                190$:
6407
6408                                    ;;*****************************************************************
6409                                    ;*TEST 26        FORMAT INVALID TRACK ADDRESS
6410                                    ;;*****************************************************************
6411  024352                            TST26:
6412  024352  000004                            SCOPE                   ;SCOPE CALL
6413  024354  000240                            NOP                     ;START OF TEST
6414  024356  012706  001100                    MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6415  024362  013700  001276                    MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6416  024366  013701  001450                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6417  024372  012737  000026  001226            MOV     #26,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6418  024400                                10$:
6419
```

```
6420                                    ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6421  024400  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
6422  024406  012737  002400  001406          MOV     #2400,RMDA0             ;TRACK = 5,SECTOR = 0
6423  024414  012737  010000  001432  15$:    MOV     #FMT16,RMOF0            ;16 BIT FORMAT
6424  024422  012737  177376  001402          MOV     #<^C<2*256.>+1>,RMWC0   ;2 * 256 WORDS
6425  024430  012737  103564  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
6426  024436  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
6427  024444  012737  001406  001174          MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
6428  024452  012737  000001  001176          MOV     #1,$TMP1
6429  024460  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
6430  024464                          20$:
6431
6432                                    ;PREPARE DEVICE FOR DATA TRANSFER
6433  024464  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6434  024470  154130                          .WORD   154130  ;TASK DESCRIPTOR
6435  024472  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
6436  024474  000240                          NOP                     ;RETURN HERE IF ERROR
6437  024476  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6438  024500  000137  025100                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6439  024504                          30$:
6440
6441                                    ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
6442  024504  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
6443  024512  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
6444  024516  112722  000006                  MOVB    #RMDA,(R2)+
6445  024522  112722  000034                  MOVB    #RMDC,(R2)+
6446  024526  112722  000032                  MOVB    #RMOF,(R2)+
6447  024532  112722  000002                  MOVB    #RMWC,(R2)+
6448  024536  112722  000004                  MOVB    #RMBA,(R2)+
6449  024542  112722  000000                  MOVB    #RMCS1,(R2)+
6450  024546  112722  000200                  MOVB    #200,(R2)+
6451  024552  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6452  024556  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
6453  024560  000240                          NOP                     ;RETURN HERE IF ERROR
6454  024562  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6455  024564  000137  025100                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6456  024570                          80$:
6457
6458                                    ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6459  024570  004737  037432                  JSR     PC,GETSTS
6460
6461                                    ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6462  024574  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6463  024600  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6464  024604  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
6465  024606  000240                          NOP                     ;RETURN HERE IF ERROR
6466  024610  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
6467  024612  000137  025100                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6468  024616                          90$:
6469
6470                                    ;VERIFY RESULTS OF WRITE COMMAND
6471  024616  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6472  024622  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
6473  024624  000240                          NOP                     ;RETURN HERE IF ERROR
6474  024626  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6475  024630  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
                        T26     FORMAT INVALID TRACK ADDRESS

6476  024632  000137  025100            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6477  024636                    100$:
6478  024636  004737  053016            JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6479  024642  000405                    BR      110$                    ;GO TO 110$ IF NO ERROR
6480  024644  000240                    NOP                     ;RETURN HERE IF ERROR
6481  024646  104000                    ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6482  024650  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6483  024652  000137  025100            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6484  024656                    110$:
6485  024656  004737  041344            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6486  024662  000405                    BR      120$                    ;GO TO 120$ IF NO ERROR
6487  024664  000240                    NOP                     ;RETURN HERE IF ERROR
6488  024666  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6489  024670  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6490  024672  000137  025100            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6491  024676                    120$:
6492
6493                            ;CLEAR "IAE" ERROR AND PREPARE DEVICE
6494  024676  004737  034000            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6495  024702  154130                    .WORD   154130  ;TASK DESCRIPTOR
6496  024704  000404                    BR      125$                    ;GO TO 125$ IF NO ERROR
6497  024706  000240                    NOP                     ;RETURN HERE IF ERROR
6498  024710  104000                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6499  024712  000137  025100            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6500  024716                    125$:
6501
6502                            ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6503  024716  012737  000073  001400        MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
6504  024724  012737  104570  001404        MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
6505  024732  004737  037766            JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6506  024736  000404                    BR      130$    ;GO TO 130$ IF NO ERROR
6507  024740  000240                    NOP                     ;RETURN HERE IF ERROR
6508  024742  104000                    ERROR                   ;ERROR DEFINED BY PUT SUB
6509  024744  000137  025100            JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6510  024750                    130$:
6511
6512                            ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6513  024750  004737  037432            JSR     PC,GETSTS
6514
6515                            ;WAIT FOR READ TO COMPLETE AND GET STATUS
6516  024754  004737  040326            JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6517  024760  004737  037516            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6518  024764  000404                    BR      140$    ;GO TO 140$ IF NO ERROR
6519  024766  000240                    NOP                     ;RETURN HERE IF ERROR
6520  024770  104000                    ERROR                   ;ERROR DEFINED BY GET SUB
6521  024772  000137  025100            JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6522  024776                    140$:
6523
6524                            ;VERIFY THE RESULTS OF READ OPERATION
6525  024776  004737  040512            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6526  025002  000405                    BR      150$                    ;GO TO 150$ IF NO ERROR
6527  025004  000240                    NOP                     ;RETURN HERE IF ERROR
6528  025006  104000                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6529  025010  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6530  025012  000137  025100            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6531  025016                    150$:
```

```
6532  025016  004737  053016          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6533  025022  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
6534  025024  000240                  NOP                     ;RETURN HERE IF ERROR
6535  025026  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6536  025030  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6537  025032  000137  025100          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6538  025036                  160$:
6539  025036  004737  041344          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6540  025042  000405                  BR      170$                    ;GO TO 170$ IF NO ERROR
6541  025044  000240                  NOP                     ;RETURN HERE IF ERROR
6542  025046  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6543  025050  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6544  025052  000137  025100          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6545  025056                  170$:
6546  025056                  180$:
6547
6548                          ;INCREMENT ADDRESS AND FORMAT NEXT TRACK
6549  025056  062737  000400  001406          ADD     #400,RMDAO              ;ADVANCE TRACK COUNT
6550  025064  022737  003400  001406          CMP     #3400,RMDAO             ;DONE ALL TRACKS??
6551  025072  103402                  BLO     190$                    ;YES!!
6552  025074  000137  024414          JMP     15$                     ;GO DO NEXT TRACK
6553  025100                  190$:
6554
6555                          ;;*******************************************************************
6556                          ;*TEST 27       FORMAT INVALID CYLINDER ADDRESS
6557                          ;;*******************************************************************
6558  025100                  TST27:
6559  025100  000004                  SCOPE                   ;SCOPE CALL
6560  025102  000240                  NOP                     ;START OF TEST
6561  025104  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6562  025110  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6563  025114  013701  001450          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6564  025120  012737  000027  001226  MOV     #27,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6565  025126                  10$:
6566
6567                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6568  025126  012737  001467  001434          MOV     #823.,RMDC0             ;START AT LAST CYLINDER + 1
6569  025134  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
6570  025142  012737  010000  001432  15$:     MOV     #FMT16,RMOF0            ;16 BIT FORMAT
6571  025150  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
6572  025156  012737  103564  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
6573  025164  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
6574  025172  012737  001434  001174          MOV     #RMDC0,$TMP0            ;USE CYLINDER FOR DATA PATTERN
6575  025200  012737  000001  001176          MOV     #1,$TMP1
6576  025206  004737  036620          JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
6577  025212                  20$:
6578
6579                          ;PREPARE DEVICE FOR DATA TRANSFER
6580  025212  004737  034000          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6581  025216  154130                  .WORD   154130  ;TASK DESCRIPTOR
6582  025220  000404                  BR      30$                     ;GO TO 30$ IF NO ERROR
6583  025222  000240                  NOP                     ;RETURN HERE IF ERROR
6584  025224  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6585  025226  000137  025626          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6586  025232                  30$:
6587
```

```
6588
6589  025232  012737  000063  001400          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
                        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
6590  025240  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
6591  025244  112722  000006                  MOVB    #RMDA,(R2)+
6592  025250  112722  000034                  MOVB    #RMDC,(R2)+
6593  025254  112722  000032                  MOVB    #RMOF,(R2)+
6594  025260  112722  000002                  MOVB    #RMWC,(R2)+
6595  025264  112722  000004                  MOVB    #RMBA,(R2)+
6596  025270  112722  000000                  MOVB    #RMCS1,(R2)+
6597  025274  112722  000200                  MOVB    #200,(R2)+
6598  025300  004737  037766                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
6599  025304  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
6600  025306  000240                          NOP                             ;RETURN HERE IF ERROR
6601  025310  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
6602  025312  000137  025626                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6603  025316                          80$:
6604
6605                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6606  025316  004737  037432                  JSR     PC,GETSTS
6607
6608                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6609  025322  004737  040326                  JSR     PC,TIMOUT                       ;WAIT FOR COMMAND TO COMPLETE
6610  025326  004737  037516                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
6611  025332  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
6612  025334  000240                          NOP                             ;RETURN HERE IF ERROR
6613  025336  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
6614  025340  000137  025626                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6615  025344                          90$:
6616
6617                                  ;VERIFY RESULTS OF WRITE COMMAND
6618  025344  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6619  025350  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
6620  025352  000240                          NOP                     ;RETURN HERE IF ERROR
6621  025354  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6622  025356  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6623  025360  000137  025626                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6624  025364                          100$:
6625  025364  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6626  025370  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
6627  025372  000240                          NOP                     ;RETURN HERE IF ERROR
6628  025374  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6629  025376  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6630  025400  000137  025626                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6631  025404                          110$:
6632  025404  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6633  025410  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
6634  025412  000240                          NOP                     ;RETURN HERE IF ERROR
6635  025414  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6636  025416  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6637  025420  000137  025626                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6638  025424                          120$:
6639
6640                                  ;CLEAR "IAE" ERROR AND PREPARE DEVICE
6641  025424  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6642  025430  154130                          .WORD   154130  ;TASK DESCRIPTOR
6643  025432  000404                          BR      125$                    ;GO TO 125$ IF NO ERROR
```

```
6644  025434  000240                          NOP                     ;RETURN HERE IF ERROR
6645  025436  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6646  025440  000137  025626                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6647  025444                          125$:
6648
6649                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6650  025444  012737  000073  001400          MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
6651  025452  012737  104570  001404          MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
6652  025460  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6653  025464  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
6654  025466  000240                          NOP                     ;RETURN HERE IF ERROR
6655  025470  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6656  025472  000137  025626                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6657  025476                          130$:
6658
6659                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6660  025476  004737  037432                  JSR     PC,GETSTS
6661
6662                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
6663  025502  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6664  025506  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6665  025512  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
6666  025514  000240                          NOP                     ;RETURN HERE IF ERROR
6667  025516  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
6668  025520  000137  025626                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6669  025524                          140$:
6670
6671                                  ;VERIFY THE RESULTS OF READ OPERATION
6672  025524  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6673  025530  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
6674  025532  000240                          NOP                     ;RETURN HERE IF ERROR
6675  025534  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6676  025536  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6677  025540  000137  025626                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6678  025544                          150$:
6679  025544  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6680  025550  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
6681  025552  000240                          NOP                     ;RETURN HERE IF ERROR
6682  025554  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6683  025556  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6684  025560  000137  025626                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6685  025564                          160$:
6686  025564  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6687  025570  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
6688  025572  000240                          NOP                     ;RETURN HERE IF ERROR
6689  025574  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6690  025576  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6691  025600  000137  025626                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6692  025604                          170$:
6693
6694                                  ;INCREMENT ADDRESS AND FORMAT NEXT CYLINDER
6695  025604  062737  000001  001434          ADD     #1,RMDCO                ;ADVANCE CYLINDER COUNT
6696  025612  022737  001777  001434          CMP     #1777,RMDCO             ;DONE ALL CYLINDERS??
6697  025620  103402                          BLO     190$                    ;YES!!
6698  025622  000137  025142                  JMP     15$                     ;GO DO NEXT SECTOR
6699  025626                          190$:
```

```
                                ;;*****************************************************************
                                ;*TEST 30       FORMAT AT OFFSET
                                ;;*****************************************************************
025626                          TST30:
025626  000004                          SCOPE                   ;SCOPE CALL
025630  000240                          NOP                     ;START OF TEST
025632  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
025636  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
025642  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
025646  012737  000030  001226          MOV     #30,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
025654                          10$:

                                ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
025654  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
025662  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
025670  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
025676  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
025704  012737  103564  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
025712  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
025720  012737  001406  001174          MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
025726  012737  000001  001176          MOV     #1,$TMP1
025734  004737  036620          15$:    JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
025740                          20$:

                                ;PREPARE DEVICE FOR DATA TRANSFER
025740  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
025744  154130                          .WORD   154130  ;TASK DESCRIPTOR
025746  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
025750  000240                          NOP                     ;RETURN HERE IF ERROR
025752  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
025754  000137  026520                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
025760                          30$:

                                ;OFFSET DEVICE FOR WRITE
025760  012737  000015  001400          MOV     #OFFSET!GO,RMCS10       ;CHANGE TO OFFSET COMMAND
025766  012702  001535                  MOV     #PUTINX,R2              ;WRITE PUT INDEX TABLE
025772  112722  000006                  MOVB    #RMDA,(R2)+
025776  112722  000034                  MOVB    #RMDC,(R2)+
026002  112722  000032                  MOVB    #RMOF,(R2)+
026006  112722  000000                  MOVB    #RMCS1,(R2)+
026012  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
026016  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
026022  000404                          BR      35$     ;GO TO 35$ IF NO ERROR
026024  000240                          NOP                     ;RETURN HERE IF ERROR
026026  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
026030  000137  026520                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
026034                          35$:

                                ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND AT OFFSET
026034  012737  000063  001400          MOV     #WH!GO,RMCS10           ; WRITE HEADER AND DATA COMMAND
026042  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
026046  112722  000002                  MOVB    #RMWC,(R2)+
026052  112722  000004                  MOVB    #RMBA,(R2)+
026056  112722  000000                  MOVB    #RMCS1,(R2)+
026062  112722  000200                  MOVB    #200,(R2)+
```

```
6756  026066  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6757  026072  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
6758  026074  000240                          NOP                     ;RETURN HERE IF ERROR
6759  026076  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6760  026100  000137  026520                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6761  026104                          80$:
6762
6763                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6764  026104  004737  037432                  JSR     PC,GETSTS
6765
6766                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6767  026110  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6768  026114  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6769  026120  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
6770  026122  000240                          NOP                     ;RETURN HERE IF ERROR
6771  026124  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
6772  026126  000137  026520                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6773  026132                          90$:
6774
6775                                  ;VERIFY RESULTS OF WRITE COMMAND
6776  026132  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6777  026136  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
6778  026140  000240                          NOP                     ;RETURN HERE IF ERROR
6779  026142  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6780  026144  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6781  026146  000137  026520                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6782  026152                          100$:
6783  026152  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6784  026156  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
6785  026160  000240                          NOP                     ;RETURN HERE IF ERROR
6786  026162  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6787  026164  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6788  026166  000137  026520                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6789  026172                          110$:
6790  026172  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6791  026176  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
6792  026200  000240                          NOP                     ;RETURN HERE IF ERROR
6793  026202  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6794  026204  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6795  026206  000137  026520                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6796  026212                          120$:
6797                                  ;OFFSET DEVICE FOR READ
6798  026212  012737  000015  001400          MOV     #OFFSET!GO,RMCS1O       ;CHANGE TO OFFSET COMMAND
6799  026220  012702  001535                  MOV     #PUTINX,R2              ;WRITE PUT INDEX TABLE
6800  026224  112722  000006                  MOVB    #RMDA,(R2)+
6801  026230  112722  000034                  MOVB    #RMDC,(R2)+
6802  026234  112722  000032                  MOVB    #RMOF,(R2)+
6803  026240  112722  000000                  MOVB    #RMCS1,(R2)+
6804  026244  112722  000200                  MOVB    #200,(R2)+
6805  026250  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6806  026254  000404                          BR      125$    ;GO TO 125$ IF NO ERROR
6807  026256  000240                          NOP                     ;RETURN HERE IF ERROR
6808  026260  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6809  026262  000137  026520                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6810  026266                          125$:
6811
```

```
6812
6813                                    ;READ HEADER AND DATA AT OFFSET
6814  026266  012737  000073  001400            MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
6815  026274  012737  104570  001404            MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
6816  026302  012702  001535                    MOV     #PUTINX,R2              ;WRITE PUT INDEX TABLE
6817  026306  112722  000002                    MOVB    #RMWC,(R2)+
6818  026312  112722  000004                    MOVB    #RMBA,(R2)+
6819  026316  112722  000000                    MOVB    #RMCS1,(R2)+
6820  026322  112722  000200                    MOVB    #200,(R2)+
6821  026326  004737  037766                    JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
6822  026332  000404                            BR      130$    ;GO TO 130$ IF NO ERROR
6823  026334  000240                            NOP                             ;RETURN HERE IF ERROR
6824  026336  104000                            ERROR                           ;ERROR DEFINED BY PUT SUB
6825  026340  000137  026520                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6826  026344                            130$:
6827
6828                                    ;WAIT FOR READ TO COMPLETE AND GET STATUS
6829  026344  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6830  026350  004737  037516                    JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
6831  026354  000404                            BR      140$    ;GO TO 140$ IF NO ERROR
6832  026356  000240                            NOP                             ;RETURN HERE IF ERROR
6833  026360  104000                            ERROR                           ;ERROR DEFINED BY GET SUB
6834  026362  000137  026520                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6835  026366                            140$:
6836
6837                                    ;VERIFY THE RESULTS OF READ OPERATION
6838  026366  004737  040512                    JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
6839  026372  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
6840  026374  000240                            NOP                             ;RETURN HERE IF ERROR
6841  026376  104000                            ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
6842  026400  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6843  026402  000137  026520                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6844  026406                            150$:
6845  026406  004737  053016                    JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6846  026412  000405                            BR      160$                    ;GO TO 160$ IF NO ERROR
6847  026414  000240                            NOP                             ;RETURN HERE IF ERROR
6848  026416  104000                            ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
6849  026420  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6850  026422  000137  026520                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6851  026426                            160$:
6852  026426  004737  041344                    JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
6853  026432  000405                            BR      170$                    ;GO TO 170$ IF NO ERROR
6854  026434  000240                            NOP                             ;RETURN HERE IF ERROR
6855  026436  104000                            ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
6856  026440  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6857  026442  000137  026520                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6858  026446                            170$:
6859
6860                                    ;VERIFY DATA
6861  026446  004737  037064                    JSR     PC,CMPBUF               ;GO COMPARE WRITE, READ DATA BUFFERS
6862  026452  103564                            .WORD   BUFONE                  ;STARTING ADDRESS OF WRITE BUFFER
6863  026454  104570                            .WORD   BUFTWO                  ;STARTING ADDDRESS OF READ BUFFER
6864  026456  000404                            BR      180$                    ;GO TO 180$ IF NO ERROR
6865  026460  000240                            NOP                             ;RETURN HERE IF ERROR
6866  026462  104000                            ERROR                           ;ERROR # DEFINED BY CMPBUF SUBROUTINE
6867  026464  000137  026520                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
```

```
6868  026470                                  180$:
6869  026470  032737  000200  001432                BIT     #OFD,RMOFO              ;DONE BOTH OFFSETS??
6870  026476  001010                                BNE     190$                    ;YES!!
6871  026500  052737  000200  001432                BIS     #OFD,RMOFO              ;SET FORWARD OFFSET
6872  026506  012737  103564  001404                MOV     #BUFONE,RMBAO           ;CHANGE DATA BUFFER
6873  026514  000137  025740                        JMP     20$
6874  026520                                  190$:
6875
6876                                          ;;*****************************************************************
6877                                          ;*TEST 31        IVC FORMAT TEST
6878                                          ;;*****************************************************************
6879  026520                                  TST31:
6880  026520  000004                                SCOPE                           ;SCOPE CALL
6881  026522  000240                                NOP                             ;START OF TEST
6882  026524  012706  001100                        MOV     #STACK,SP               ;INITIALIZE STACK POINTER
6883  026530  013700  001276                        MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
6884  026534  013701  001450                        MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
6885  026540  012737  000031  001226                MOV     #31,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
6886  026546                                  10$:
6887
6888                                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6889  026546  012737  000000  001434                MOV     #0,RMDC0                ;CYLINDER = 0
6890  026554  012737  000000  001406                MOV     #0,RMDA0                ;TRACK = SECTOR = 0
6891  026562  012737  010000  001432                MOV     #FMT16,RMOFO            ;16 BIT FORMAT
6892  026570  012737  177376  001402                MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
6893  026576  012737  103564  001404                MOV     #BUFONE,RMBAO           ;DATA BUFFER ADDRESS
6894  026604  012737  000062  001400                MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
6895  026612  012737  001406  001174                MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
6896  026620  012737  000001  001176                MOV     #1,$TMP1
6897  026626  004737  036620                15$:    JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
6898  026632                                  20$:
6899
6900                                          ;PREPARE DEVICE FOR DATA TRANSFER
6901  026632  004737  034000                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6902  026636  154130                                .WORD   154130  ;TASK DESCRIPTOR
6903  026640  000404                                BR      30$                     ;GO TO 30$ IF NO ERROR
6904  026642  000240                                NOP                     ;RETURN HERE IF ERROR
6905  026644  104000                                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6906  026646  000137  027326                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6907  026652                                  30$:
6908
6909                                          ;RESET VOLUME VALID BY SETTING AND RESETTING DIAGNOSTIC MODE
6910  026652  012737  000001  001424                MOV     #DMD,RMMR10             ;SET DIAGNOSTIC MODE
6911  026660  112737  000024  001535                MOVB    #RMMR1,PUTINX           ;WRITE REGISTER INDEX TABLE
6912  026666  112737  000200  001536                MOVB    #200,PUTINX+1
6913  026674  004737  037766                        JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6914  026700  000404                                BR      35$     ;GO TO 35$ IF NO ERROR
6915  026702  000240                                NOP                     ;RETURN HERE IF ERROR
6916  026704  104000                                ERROR                   ;ERROR DEFINED BY PUT SUB
6917  026706  000137  027326                        JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6918  026712                                  35$:
6919
6920                                          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
6921  026712  012737  000063  001400                MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
6922  026720  012737  000000  001424                MOV     #0,RMMR10               ;RESET DIAGNOSTIC MODE
6923  026726  012702  001535                        MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
```

```
6924  026732  112722  000024                  MOVB    #RMMR1,(R2)+
6925  026736  112722  000006                  MOVB    #RMDA,(R2)+
6926  026742  112722  000034                  MOVB    #RMDC,(R2)+
6927  026746  112722  000032                  MOVB    #RMOF,(R2)+
6928  026752  112722  000002                  MOVB    #RMWC,(R2)+
6929  026756  112722  000004                  MOVB    #RMBA,(R2)+
6930  026762  112722  000000                  MOVB    #RMCS1,(R2)+
6931  026766  112722  000200                  MOVB    #200,(R2)+
6932  026772  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6933  026776  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
6934  027000  000240                          NOP                     ;RETURN HERE IF ERROR
6935  027002  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6936  027004  000137  027326                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6937  027010                          80$:
6938
6939                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6940  027010  004737  037432                  JSR     PC,GETSTS
6941
6942                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6943  027014  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6944  027020  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6945  027024  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
6946  027026  000240                          NOP                     ;RETURN HERE IF ERROR
6947  027030  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
6948  027032  000137  027326                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6949  027036                          90$:
6950
6951                                  ;VERIFY RESULTS OF WRITE COMMAND
6952  027036  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6953  027042  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
6954  027044  000240                          NOP                     ;RETURN HERE IF ERROR
6955  027046  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6956  027050  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6957  027052  000137  027326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6958  027056                          100$:
6959  027056  032737  010000  001372          BIT     #IVC,RMER21             ;IS IVC STATUS SET??
6960  027064  001012                          BNE     110$                    ;YES!!
6961  027066  013737  001372  001142          MOV     RMER21,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
6962  027074  013737  001372  001140          MOV     RMER21,$GDDAT           ;EXPECTED STATUS
6963  027102  052737  010000  001140          BIS     #IVC,$GDDAT
6964  027110  104342                          ERROR   342                     ;IVC NOT SET DURING FORMAT
6965  027112                          110$:
6966  027112  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6967  027116  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
6968  027120  000240                          NOP                     ;RETURN HERE IF ERROR
6969  027122  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6970  027124  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6971  027126  000137  027326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6972  027132                          120$:
6973
6974                                  ;CLEAR IVC ERROR
6975  027132  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6976  027136  040100                          .WORD   040100  ;TASK DESCRIPTOR
6977  027140  000404                          BR      125$                    ;GO TO 125$ IF NO ERROR
6978  027142  000240                          NOP                     ;RETURN HERE IF ERROR
6979  027144  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
6980  027146  000137  027326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6981  027152                              125$:
6982
6983                                      ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6984  027152  012737  000073  001400          MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
6985  027160  012737  104570  001404          MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
6986  027166  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6987  027172  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
6988  027174  000240                          NOP                     ;RETURN HERE IF ERROR
6989  027176  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6990  027200  000137  027326                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6991  027204                              130$:
6992
6993                                      ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6994  027204  004737  037432                  JSR     PC,GETSTS
6995
6996                                      ;WAIT FOR READ TO COMPLETE AND GET STATUS
6997  027210  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6998  027214  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6999  027220  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
7000  027222  000240                          NOP                     ;RETURN HERE IF ERROR
7001  027224  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7002  027226  000137  027326                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
7003  027232                              140$:
7004
7005                                      ;VERIFY THE RESULTS OF READ OPERATION
7006  027232  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7007  027236  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
7008  027240  000240                          NOP                     ;RETURN HERE IF ERROR
7009  027242  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7010  027244  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7011  027246  000137  027326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
7012  027252                              150$:
7013  027252  032737  010000  001372          BIT     #IVC,RMER21             ;IS IVC STATUS SET??
7014  027260  001012                          BNE     160$                    ;YES!!
7015  027262  013737  001372  001142          MOV     RMER21,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
7016  027270  013737  001372  001140          MOV     RMER21,$GDDAT           ;EXPECTED STATUS
7017  027276  052737  010000  001140          BIS     #IVC,$GDDAT
7018  027304  104342                          ERROR   342                     ;IVC NOT SET DURING FORMAT
7019  027306                              160$:
7020  027306  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7021  027312  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
7022  027314  000240                          NOP                     ;RETURN HERE IF ERROR
7023  027316  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7024  027320  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7025  027322  000137  027326                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
7026  027326                              170$:
7027  027326                              190$:
7028
7029                                      ;;*******************************************************************
7030                                      ;*TEST 32       FORMAT ERROR TEST - 18
7031                                      ;;*******************************************************************
7032  027326                              TST32:
7033  027326  000004                          SCOPE                   ;SCOPE CALL
7034  027330  000240                          NOP                     ;START OF TEST
7035  027332  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
```

```
7036  027336  013700  001276              MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7037  027342  013701  001450              MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7038  027346  012737  000032  001226      MOV     #32,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7039  027354                          10$:
7040
7041                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
7042  027354  012737  000000  001434      MOV     #0,RMDC0                ;CYLINDER = 0
7043  027362  012737  000000  001406      MOV     #0,RMDA0                ;TRACK = SECTOR = 0
7044  027370  012737  000000  001432      MOV     #0,RMOF0                ;18 BIT FORMAT
7045  027376  012737  177376  001402      MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
7046  027404  012737  103564  001404      MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
7047  027412  012737  000062  001400      MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
7048  027420  012737  065104  001174      MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
7049  027426  012737  000001  001176      MOV     #1,$TMP1
7050  027434  004737  036620              JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
7051  027440                          20$:
7052
7053                                  ;PREPARE DEVICE FOR DATA TRANSFER
7054  027440  004737  034000              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7055  027444  154130                      .WORD   154130  ;TASK DESCRIPTOR
7056  027446  000404                      BR      30$                     ;GO TO 30$ IF NO ERROR
7057  027450  000240                      NOP                     ;RETURN HERE IF ERROR
7058  027452  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7059  027454  000137  030050              JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7060  027460                          30$:
7061
7062                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
7063  027460  012737  000063  001400      MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
7064  027466  012702  001535              MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
7065  027472  112722  000006              MOVB    #RMDA,(R2)+
7066  027476  112722  000034              MOVB    #RMDC,(R2)+
7067  027502  112722  000032              MOVB    #RMOF,(R2)+
7068  027506  112722  000002              MOVB    #RMWC,(R2)+
7069  027512  112722  000004              MOVB    #RMBA,(R2)+
7070  027516  112722  000000              MOVB    #RMCS1,(R2)+
7071  027522  112722  000200              MOVB    #200,(R2)+
7072  027526  004737  037766              JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7073  027532  000404                      BR      80$     ;GO TO 80$ IF NO ERROR
7074  027534  000240                      NOP                     ;RETURN HERE IF ERROR
7075  027536  104000                      ERROR                   ;ERROR DEFINED BY PUT SUB
7076  027540  000137  030050              JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7077  027544                          80$:
7078
7079                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
7080  027544  004737  037432              JSR     PC,GETSTS
7081
7082                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
7083  027550  004737  040326              JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
7084  027554  004737  037516              JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7085  027560  000404                      BR      90$     ;GO TO 90$ IF NO ERROR
7086  027562  000240                      NOP                     ;RETURN HERE IF ERROR
7087  027564  104000                      ERROR                   ;ERROR DEFINED BY GET SUB
7088  027566  000137  030050              JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7089  027572                          90$:
7090
7091                                  ;VERIFY RESULTS OF WRITE COMMAND
```

```
7092  027572  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7093  027576  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
7094  027600  000240                          NOP                     ;RETURN HERE IF ERROR
7095  027602  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7096  027604  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7097  027606  000137  030050                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7098  027612                          100$:
7099  027612  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7100  027616  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
7101  027620  000240                          NOP                     ;RETURN HERE IF ERROR
7102  027622  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7103  027624  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7104  027626  000137  030050                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7105  027632                          110$:
7106  027632  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7107  027636  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
7108  027640  000240                          NOP                     ;RETURN HERE IF ERROR
7109  027642  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7110  027644  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7111  027646  000137  030050                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7112  027652                          120$:
7113
7114
7115                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN AT 16 BIT FORMAT
7116  027652  012737  010000  001432          MOV     #FMT16,RMOF0            ;CHANGE TO 16 BIT FORMAT
7117  027660  012737  000073  001400          MOV     #RH!GO,RMCS1            ;READ HEADER & DATA COMMAND
7118  027666  012737  104570  001404          MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
7119  027674  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7120  027700  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
7121  027702  000240                          NOP                     ;RETURN HERE IF ERROR
7122  027704  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
7123  027706  000137  030050                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7124  027712                          130$:
7125
7126                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
7127  027712  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
7128  027716  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7129  027722  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
7130  027724  000240                          NOP                     ;RETURN HERE IF ERROR
7131  027726  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7132  027730  000137  030050                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7133  027734                          140$:
7134
7135                                  ;VERIFY THE RESULTS OF READ OPERATION
7136  027734  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7137  027740  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
7138  027742  000240                          NOP                     ;RETURN HERE IF ERROR
7139  027744  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7140  027746  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7141  027750  000137  030050                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7142  027754                          150$:
7143
7144                                  ;VERIFY THAT FORMAT ERROR IS SET
7145  027754  032737  000020  001344          BIT     #FER,RMER1              ;IS FORMAT ERROR SET??
7146  027762  001022                          BNE     160$                    ;YES!!
7147  027764  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
```

```
7148  027770  000405                          BR      155$                    ;GO TO 155$ IF NO ERROR
7149  027772  000240                          NOP                     ;RETURN HERE IF ERROR
7150  027774  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7151  027776  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7152  030000  000137  030050                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7153  030004  013737  001344  001142  155$:   MOV     RMER11,$BDDAT           ;BAD DATA FOR TYPEOUT
7154  030012  013737  001344  001140          MOV     RMER11,$GDDAT           ;EXPECTED DATA
7155  030020  052737  000020  001140          BIS     #FER,$GDDAT
7156  030026  104343                          ERROR   343                     ;FORMAT ERROR NOT SET
7157  030030                          160$:
7158  030030  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7159  030034  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
7160  030036  000240                          NOP                     ;RETURN HERE IF ERROR
7161  030040  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7162  030042  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7163  030044  000137  030050                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7164  030050                          170$:
7165  030050                          180$:
7166
7167                                  ;;*******************************************************************
7168                                  ;*TEST 33        FORMAT ERROR TEST - 16
7169                                  ;;*******************************************************************
7170  030050                          TST33:
7171  030050  000004                          SCOPE                   ;SCOPE CALL
7172  030052  000240                          NOP                     ;START OF TEST
7173  030054  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7174  030060  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7175  030064  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7176  030070  012737  000033  001226          MOV     #33,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7177  030076                          10$:
7178
7179                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
7180  030076  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
7181  030104  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
7182  030112  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7183  030120  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
7184  030126  012737  103564  001404          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
7185  030134  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
7186  030142  012737  065104  001174          MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
7187  030150  012737  000001  001176          MOV     #1,$TMP1
7188  030156  004737  036620                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
7189  030162                          20$:
7190
7191                                  ;PREPARE DEVICE FOR DATA TRANSFER
7192  030162  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7193  030166  154130                          .WORD   154130  ;TASK DESCRIPTOR
7194  030170  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
7195  030172  000240                          NOP                     ;RETURN HERE IF ERROR
7196  030174  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7197  030176  000137  030572                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7198  030202                          30$:
7199
7200                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
7201  030202  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
7202  030210  012702  001535                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
7203  030214  112722  000006                  MOVB    #RMDA,(R2)+
```

```
7204  030220  112722  000034                  MOVB    #RMDC,(R2)+
7205  030224  112722  000032                  MOVB    #RMOF,(R2)+
7206  030230  112722  000002                  MOVB    #RMWC,(R2)+
7207  030234  112722  000004                  MOVB    #RMBA,(R2)+
7208  030240  112722  000000                  MOVB    #RMCS1,(R2)+
7209  030244  112722  000200                  MOVB    #200,(R2)+
7210  030250  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7211  030254  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
7212  030256  000240                          NOP                     ;RETURN HERE IF ERROR
7213  030260  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
7214  030262  000137  030572                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7215  030266                          80$:
7216
7217                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
7218  030266  004737  037432                  JSR     PC,GETSTS
7219
7220                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
7221  030272  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
7222  030276  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7223  030302  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
7224  030304  000240                          NOP                     ;RETURN HERE IF ERROR
7225  030306  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7226  030310  000137  030572                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7227  030314                          90$:
7228
7229                                  ;VERIFY RESULTS OF WRITE COMMAND
7230  030314  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7231  030320  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
7232  030322  000240                          NOP                     ;RETURN HERE IF ERROR
7233  030324  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7234  030326  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7235  030330  000137  030572                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7236  030334                          100$:
7237  030334  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7238  030340  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
7239  030342  000240                          NOP                     ;RETURN HERE IF ERROR
7240  030344  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7241  030346  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7242  030350  000137  030572                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7243  030354                          110$:
7244  030354  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7245  030360  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
7246  030362  000240                          NOP                     ;RETURN HERE IF ERROR
7247  030364  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7248  030366  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7249  030370  000137  030572                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7250  030374                          120$:
7251
7252
7253                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN AT 18 BIT FORMAT
7254  030374  012737  000000  001432            MOV     #0,RMOF0                ;SET FOR 18 BIT FORMAT
7255  030402  012737  000073  001400            MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
7256  030410  012737  104570  001404            MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
7257  030416  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7258  030422  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
7259  030424  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
7260  030426  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
7261  030430  000137  030572                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7262  030434                        130$:
7263
7264                                ;WAIT FOR READ TO COMPLETE AND GET STATUS
7265  030434  004737  040326                JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
7266  030440  004737  037516                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7267  030444  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
7268  030446  000240                        NOP                     ;RETURN HERE IF ERROR
7269  030450  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
7270  030452  000137  030572                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7271  030456                        140$:
7272
7273                                ;VERIFY THE RESULTS OF READ OPERATION
7274  030456  004737  040512                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7275  030462  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
7276  030464  000240                        NOP                     ;RETURN HERE IF ERROR
7277  030466  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7278  030470  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7279  030472  000137  030572                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7280  030476                        150$:
7281
7282                                ;VERIFY THAT FORMAT ERROR IS SET
7283  030476  032737  000020  001344        BIT     #FER,RMER11             ;IS FORMAT ERROR SET??
7284  030504  001022                        BNE     160$                    ;YES!!
7285  030506  004737  053016                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7286  030512  000405                        BR      155$                    ;GO TO 155$ IF NO ERROR
7287  030514  000240                        NOP                     ;RETURN HERE IF ERROR
7288  030516  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7289  030520  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7290  030522  000137  030572                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7291  030526  013737  001344  001142  155$:  MOV     RMER11,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
7292  030534  013737  001344  001140        MOV     RMER11,$GDDAT           ;EXPECTED STATUS
7293  030542  052737  000020  001140        BIS     #FER,$GDDAT
7294  030550  104343                        ERROR   343                     ;FORMAT ERROR NOT SET
7295  030552                        160$:
7296  030552  004737  041344                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7297  030556  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
7298  030560  000240                        NOP                     ;RETURN HERE IF ERROR
7299  030562  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7300  030564  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7301  030566  000137  030572                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7302  030572                        170$:
7303  030572                        180$:
7304
7305                                ;;*******************************************************************
7306                                ;*TEST 34        FORMAT HCE, FIRST HEADER WORD
7307                                ;;*******************************************************************
7308  030572                        TST34:
7309  030572  000004                        SCOPE                   ;SCOPE CALL
7310  030574  000240                        NOP                     ;START OF TEST
7311  030576  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7312  030602  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7313  030606  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7314  030612  012737  000034  001226        MOV     #34,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7315  030620  012737  000001  031576        MOV     #1,190$                 ;INIT BIT POSITION
```

```
7316  030626                                  10$:
7317
7318                                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
7319  030626  012737  000000  001434                MOV     #0,RMDC0                ;CYLINDER = 0
7320  030634  012737  000000  001406                MOV     #0,RMDA0                ;TRACK = SECTOR = 0
7321  030642  012737  010000  001432                MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7322  030650  012737  177376  001402                MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
7323  030656  012737  103564  001404                MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
7324  030664  012737  000062  001400                MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
7325  030672  012737  065104  001174                MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
7326  030700  012737  000001  001176                MOV     #1,$TMP1
7327  030706  004737  036620                      JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
7328  030712                                  20$:
7329
7330                                          ;PREPARE DEVICE FOR DATA TRANSFER
7331  030712  004737  034000                      JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7332  030716  154130                              .WORD   154130  ;TASK DESCRIPTOR
7333  030720  000404                              BR      30$                     ;GO TO 30$ IF NO ERROR
7334  030722  000240                              NOP                     ;RETURN HERE IF ERROR
7335  030724  104000                              ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7336  030726  000137  031750                      JMP     240$                    ;GO TO 240$ IF ERROR WAS FOUND
7337  030732                                  30$:
7338
7339                                          ;COMPLEMENT DATA BIT IN FIRST HEADER WORD
7340  030732  033737  031576  103564                BIT     190$,BUFONE             ;IS BIT IN HEADER ON??
7341  030740  001404                              BEQ     31$                     ;NO!!
7342  030742  043737  031576  103564                BIC     190$,BUFONE             ;RESET BIT IN HEADER
7343  030750  000403                              BR      32$
7344  030752  053737  031576  103564  31$:         BIS     190$,BUFONE             ;SET BIT IN HEADER
7345
7346  030760                                  32$:
7347
7348                                          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
7349  030760  012737  000063  001400                MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
7350  030766  012702  001535                      MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
7351  030772  112722  000006                      MOVB    #RMDA,(R2)+
7352  030776  112722  000034                      MOVB    #RMDC,(R2)+
7353  031002  112722  000032                      MOVB    #RMOF,(R2)+
7354  031006  112722  000002                      MOVB    #RMWC,(R2)+
7355  031012  112722  000004                      MOVB    #RMBA,(R2)+
7356  031016  112722  000000                      MOVB    #RMCS1,(R2)+
7357  031022  112722  000200                      MOVB    #200,(R2)+
7358  031026  004737  037766                      JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7359  031032  000404                              BR      80$     ;GO TO 80$ IF NO ERROR
7360  031034  000240                              NOP                     ;RETURN HERE IF ERROR
7361  031036  104000                              ERROR                   ;ERROR DEFINED BY PUT SUB
7362  031040  000137  031574                      JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7363  031044                                  80$:
7364
7365                                          ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
7366  031044  004737  037432                      JSR     PC,GETSTS
7367
7368                                          ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
7369  031050  004737  040326                      JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
7370  031054  004737  037516                      JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7371  031060  000404                              BR      90$     ;GO TO 90$ IF NO ERROR
```

```
7372  031062  000240                    T34     FORMAT HCE, FIRST HEADER WORD
7373  031064  104000                            NOP                     ;RETURN HERE IF ERROR
7374  031066  000137  031574                    ERROR                   ;ERROR DEFINED BY GET SUB
7375  031072                                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7376                                    90$:
7377
7378  031072  004737  040512            ;VERIFY RESULTS OF WRITE COMMAND
7379  031076  000405                            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7380  031100  000240                            BR      100$                    ;GO TO 100$ IF NO ERROR
7381  031102  104000                            NOP                     ;RETURN HERE IF ERROR
7382  031104  004736                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7383  031106  000137  031574                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7384  031112                                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7385  031112  004737  053016            100$:
7386  031116  000405                            JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7387  031120  000240                            BR      110$                    ;GO TO 110$ IF NO ERROR
7388  031122  104000                            NOP                     ;RETURN HERE IF ERROR
7389  031124  004736                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7390  031126  000137  031574                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7391  031132                                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7392  031132  004737  041344            110$:
7393  031136  000405                            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7394  031140  000240                            BR      120$                    ;GO TO 120$ IF NO ERROR
7395  031142  104000                            NOP                     ;RETURN HERE IF ERROR
7396  031144  004736                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7397  031146  000137  031574                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7398  031152                                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7399                                    120$:
7400
7401                                    ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
7402  031152  012737  000073  001400            MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
7403  031160  012737  104570  001404            MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
7404  031166  004737  037766                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7405  031172  000404                            BR      130$    ;GO TO 130$ IF NO ERROR
7406  031174  000240                            NOP                     ;RETURN HERE IF ERROR
7407  031176  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
7408  031200  000137  031574                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7409  031204                                    130$:
7410
7411                                    ;WAIT FOR READ TO COMPLETE AND GET STATUS
7412  031204  004737  040326                    JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
7413  031210  004737  037516                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7414  031214  000404                            BR      140$    ;GO TO 140$ IF NO ERROR
7415  031216  000240                            NOP                     ;RETURN HERE IF ERROR
7416  031220  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
7417  031222  000137  031574                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7418  031226                                    140$:
7419
7420                                    ;VERIFY THE RESULTS OF READ OPERATION
7421  031226  004737  040512                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7422  031232  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
7423  031234  000240                            NOP                     ;RETURN HERE IF ERROR
7424  031236  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7425  031240  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7426  031242  000137  031574                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7427  031246                                    150$:
```

```
7428  031246  032737  140000  031576          BIT     #MSE!USE,190$           ;IS THIS BSE ??
7429  031254  001033                          BNE     152$                    ;YES !!
7430  031256  032737  010000  031576          BIT     #FMT16,190$             ;IS THIS FER ??
7431  031264  001466                          BEQ     155$                    ;NO !!
7432
7433                                  ;VERIFY THAT FER IS SET
7434  031266  032737  000020  001344          BIT     #FER,RMER1I             ;IS FER SET ??
7435  031274  001122                          BNE     160$                    ;YES !!
7436  031276  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7437  031302  000405                          BR      151$                    ;GO TO 151$ IF NO ERROR
7438  031304  000240                          NOP                     ;RETURN HERE IF ERROR
7439  031306  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7440  031310  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7441  031312  000137  031574                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7442  031316  013737  001344  001142  151$:   MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
7443  031324  013737  001344  001140          MOV     RMER1I,$GDDAT           ;EXPECTED STATUS
7444  031332  052737  000020  001140          BIS     #FER,$GDDAT
7445  031340  104343                          ERROR   343                     ;FORMAT ERROR NOT SET
7446  031342  000514                          BR      180$
7447  031344                          152$:
7448                                  ;VERIFY THAT BAD SECTOR ERROR IS SET
7449  031344  032737  100000  001372          BIT     #BSE,RMER2I             ;IS BSE SET ??
7450  031352  001073                          BNE     160$                    ;YES !!
7451  031354  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7452  031360  000405                          BR      153$                    ;GO TO 153$ IF NO ERROR
7453  031362  000240                          NOP                     ;RETURN HERE IF ERROR
7454  031364  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7455  031366  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7456  031370  000137  031574                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7457  031374                          153$:
7458  031374  013737  001372  001142          MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
7459  031402  013737  001372  001140          MOV     RMER2I,$GDDAT           ;EXPECTED STATUS
7460  031410  052737  100000  001140          BIS     #BSE,$GDDAT
7461  031416  013737  103564  001174          MOV     BUFONE,$TMP0            ;WRITTEN HEADER
7462  031424  013737  104570  001176          MOV     BUFTWO,$TMP1            ;READ HEADER WORD
7463  031432  013737  031576  001200          MOV     190$,$TMP2              ;FAILING BIT POSITION
7464                                  ;       ERROR   345                     ;FAILED TO DETECT BSE
7465  031440  000455                          BR      180$                    ;SKIP REST OF TEST
7466  031442                          155$:
7467
7468                                  ;VERIFY THAT HEADER COMPARE ERROR IS SET
7469  031442  032737  000200  001344          BIT     #HCE,RMER1I             ;IS "HCE" SET??
7470  031450  001034                          BNE     160$                    ;YES!!
7471  031452  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7472  031456  000405                          BR      156$                    ;GO TO 156$ IF NO ERROR
7473  031460  000240                          NOP                     ;RETURN HERE IF ERROR
7474  031462  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7475  031464  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7476  031466  000137  031574                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7477  031472  013737  001344  001142  156$:   MOV     RMER1I,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
7478  031500  013737  001344  001140          MOV     RMER1I,$GDDAT           ;EXPECTED STATUS
7479  031506  052737  000200  001140          BIS     #HCE,$GDDAT
7480  031514  013737  103564  001174          MOV     BUFONE,$TMP0            ;WRITTEN HEADER
7481  031522  013737  104570  001176          MOV     BUFTWO,$TMP1            ;READ HEADER WORD
7482  031530  013737  031576  001200          MOV     190$,$TMP2              ;FAILING BIT POSITION
7483  031536  104344                          ERROR   344                     ;FORMAT ERROR NOT SET
```

```
7484  031540  000415                          BR      180$
7485
7486  031542                          160$:
7487
7488                                  ;CHECK FOR OTHER ERRORS
7489  031542  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7490  031546  000405                          BR      165$                    ;GO TO 165$ IF NO ERROR
7491  031550  000240                          NOP                     ;RETURN HERE IF ERROR
7492  031552  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7493  031554  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7494  031556  000137  031574                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7495  031562                          165$:
7496
7497
7498
7499                                  ;ADVANCE THE BIT POSITION AND FORMAT AGAIN IF NOT DONE
7500  031562  006337  031576                  ASL     190$                    ;SHIFT TO NEXT BIT POSITION
7501  031566  001404                          BEQ     200$                    ;EXIT IF DONE
7502  031570  000137  030626                  JMP     10$                     ;GO DO NEXT SECTOR
7503  031574                          180$:
7504  031574  000465                          BR      240$                    ;JUMP OVER STORAGE
7505  031576                          190$:
7506  031576  000000                          .WORD                   ;STORAGE FOR BIT POSITION
7507  031600                          200$:
7508                                  ;       MOV     #0,RMDC0                ;REFORMAT THE SECTOR IN 16 BIT MODE
7509                                  ;       MOV     #0,RMDA0                ;
7510  031600  012737  010000  001432          MOV     #FMT16,RMOF0            ; 16 BIT MODE
7511  031606  012737  177776  001402          MOV     #-2,RMWC0               ;ONLY TWO HEADER WORDS
7512  031614  012737  103564  001404          MOV     #BUFONE,RMBA0           ;SELECT THE BUFFER
7513  031622  012737  000062  001400          MOV     #WH,RMCS10              ;WRITE HEAD AND DATA COMMAND
7514  031630  004737  036620                  JSR     PC,GENBUF               ;GENERATE THE BUFFER PATTERN
7515
7516  031634  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7517  031640  154130                          .WORD   154130  ;TASK DESCRIPTOR
7518  031642  000404                          BR      210$                    ;GO TO 210$ IF NO ERROR
7519  031644  000240                          NOP                     ;RETURN HERE IF ERROR
7520  031646  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7521  031650  000137  031654                  JMP     210$                    ;GO TO 210$ IF ERROR WAS FOUND
7522  031654  012737  000063  001400  210$:   MOV     #WH!GO,RMCS10           ;SET THE GO BIT
7523  031662  012702  001535                  MOV     #PUTINX,R2              ;PUT THE REGISTER ADDRESS INTO TABLE
7524  031666  112722  000006                  MOVB    #RMDA,(R2)+             ;
7525  031672  112722  000034                  MOVB    #RMDC,(R2)+             ;
7526  031676  112722  000032                  MOVB    #RMOF,(R2)+             ;
7527  031702  112722  000002                  MOVB    #RMWC,(R2)+             ;
7528  031706  112722  000004                  MOVB    #RMBA,(R2)+             ;
7529  031712  112722  000000                  MOVB    #RMCS1,(R2)+            ;
7530  031716  112722  000200                  MOVB    #200,(R2)+              ;TERMINATOR
7531  031722  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7532  031726  000404                          BR      220$    ;GO TO 220$ IF NO ERROR
7533  031730  000240                          NOP                     ;RETURN HERE IF ERROR
7534  031732  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
7535  031734  000137  031750                  JMP     240$    ;GO TO 240$ IF ERROR WAS FOUND
7536  031740  000240                  220$:   NOP
7537  031742  004737  040326                  JSR     PC,TIMOUT               ;WAOT UNTIL IT FINISH
7538  031746  000240                          NOP
7539  031750                          240$:
```

```
7540
7541                                    ;;*****************************************************************
7542                                    ;*TEST 35       FORMAT HCE, SECOND HEADER WORD
7543                                    ;;*****************************************************************
7544  031750                            TST35:
7545  031750  000004                           SCOPE                   ;SCOPE CALL
7546  031752  000240                           NOP                     ;START OF TEST
7547  031754  012706  001100                   MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7548  031760  013700  001276                   MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7549  031764  013701  001450                   MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7550  031770  012737  000035  001226           MOV     #35,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7551  031776  012737  000001  032560           MOV     #1,190$                 ;INIT BIT POSITION
7552  032004                            10$:
7553
7554                                    ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
7555  032004  012737  000000  001434           MOV     #0,RMDC0                ;CYLINDER = 0
7556  032012  012737  000000  001406           MOV     #0,RMDA0                ;TRACK = SECTOR = 0
7557  032020  012737  010000  001432           MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7558  032026  012737  177376  001402           MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
7559  032034  012737  103564  001404           MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
7560  032042  012737  000062  001400           MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
7561  032050  012737  065104  001174           MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
7562  032056  012737  000001  001176           MOV     #1,$TMP1
7563  032064  004737  036620                   JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
7564  032070                            20$:
7565
7566                                    ;PREPARE DEVICE FOR DATA TRANSFER
7567  032070  004737  034000                   JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7568  032074  154130                           .WORD   154130  ;TASK DESCRIPTOR
7569  032076  000404                           BR      30$                     ;GO TO 30$ IF NO ERROR
7570  032100  000240                           NOP                     ;RETURN HERE IF ERROR
7571  032102  104000                           ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7572  032104  000137  032732                   JMP     240$                    ;GO TO 240$ IF ERROR WAS FOUND
7573  032110                            30$:
7574
7575                                    ;COMPLEMENT DATA BIT IN SECOND HEADER WORD
7576  032110  033737  032560  103566           BIT     190$,BUFONE+2           ;IS BIT IN HEADER ON??
7577  032116  001404                           BEQ     31$                     ;NO!!
7578  032120  043737  032560  103566           BIC     190$,BUFONE+2           ;RESET BIT IN HEADER
7579  032126  000403                           BR      32$
7580  032130  053737  032560  103566   31$:    BIS     190$,BUFONE+2           ;SET BIT IN HEADER
7581
7582  032136                            32$:
7583
7584                                    ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
7585  032136  012737  000063  001400           MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
7586  032144  012702  001535                   MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
7587  032150  112722  000006                   MOVB    #RMDA,(R2)+
7588  032154  112722  000034                   MOVB    #RMDC,(R2)+
7589  032160  112722  000032                   MOVB    #RMOF,(R2)+
7590  032164  112722  000002                   MOVB    #RMWC,(R2)+
7591  032170  112722  000004                   MOVB    #RMBA,(R2)+
7592  032174  112722  000000                   MOVB    #RMCS1,(R2)+
7593  032200  112722  000200                   MOVB    #200,(R2)+
7594  032204  004737  037766                   JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7595  032210  000404                           BR      80$     ;GO TO 80$ IF NO ERROR
```

```
7596  032212  000240                          NOP                     ;RETURN HERE IF ERROR
7597  032214  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
7598  032216  000137  032556                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7599  032222                          80$:
7600
7601                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
7602  032222  004737  037432                  JSR     PC,GETSTS
7603
7604                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
7605  032226  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
7606  032232  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7607  032236  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
7608  032240  000240                          NOP                     ;RETURN HERE IF ERROR
7609  032242  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7610  032244  000137  032556                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7611  032250                          90$:
7612
7613                                  ;VERIFY RESULTS OF WRITE COMMAND
7614  032250  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7615  032254  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
7616  032256  000240                          NOP                     ;RETURN HERE IF ERROR
7617  032260  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7618  032262  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7619  032264  000137  032556                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7620  032270                          100$:
7621  032270  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7622  032274  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
7623  032276  000240                          NOP                     ;RETURN HERE IF ERROR
7624  032300  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7625  032302  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7626  032304  000137  032556                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7627  032310                          110$:
7628  032310  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7629  032314  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
7630  032316  000240                          NOP                     ;RETURN HERE IF ERROR
7631  032320  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7632  032322  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7633  032324  000137  032556                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7634  032330                          120$:
7635
7636
7637                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
7638  032330  012737  000073  001400            MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
7639  032336  012737  104570  001404            MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
7640  032344  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7641  032350  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
7642  032352  000240                          NOP                     ;RETURN HERE IF ERROR
7643  032354  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
7644  032356  000137  032556                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7645  032362                          130$:
7646
7647                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
7648  032362  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
7649  032366  004737  037516                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7650  032372  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
7651  032374  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
7652  032376  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
7653  032400  000137  032556                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7654  032404                          140$:
7655
7656                                  ;VERIFY THE RESULTS OF READ OPERATION
7657  032404  004737  040512                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7658  032410  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
7659  032412  000240                          NOP                     ;RETURN HERE IF ERROR
7660  032414  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7661  032416  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7662  032420  000137  032556                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7663  032424                          150$:
7664
7665                                  ;VERIFY THAT HEADER COMPARE ERROR IS SET
7666  032424  032737  000200  001344          BIT     #HCE,RMER11             ;IS "HCE" SET??
7667  032432  001034                          BNE     160$                    ;YES!!
7668  032434  004737  053016                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7669  032440  000405                          BR      155$                    ;GO TO 155$ IF NO ERROR
7670  032442  000240                          NOP                     ;RETURN HERE IF ERROR
7671  032444  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7672  032446  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7673  032450  000137  032556                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7674  032454  013737  001344  001142  155$:   MOV     RMER11,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
7675  032462  013737  001344  001140          MOV     RMER11,$GDDAT           ;EXPECTED STATUS
7676  032470  052737  000200  001140          BIS     #HCE,$GDDAT
7677  032476  013737  103566  001174          MOV     BUFONE+2,$TMP0          ;WRITTEN HEADER
7678  032504  013737  104572  001176          MOV     BUFTWO+2,$TMP1          ;READ HEADER WORD
7679  032512  013737  032560  001200          MOV     190$,$TMP2              ;FAILING BIT POSITION
7680  032520  104344                          ERROR   344                     ;FORMAT ERROR NOT SET
7681  032522  000415                          BR      180$
7682
7683  032524                          160$:
7684
7685                                  ;CHECK FOR OTHER ERRORS
7686  032524  004737  041344                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7687  032530  000405                          BR      165$                    ;GO TO 165$ IF NO ERROR
7688  032532  000240                          NOP                     ;RETURN HERE IF ERROR
7689  032534  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7690  032536  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7691  032540  000137  032556                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7692  032544                          165$:
7693
7694
7695                                  ;ADVANCE THE BIT POSITION AND FORMAT NEXT SECTOR IF NOT DONE
7696  032544  006337  032560                  ASL     190$                    ;SHIFT TO NEXT BIT POSITION
7697  032550  001404                          BEQ     200$                    ;EXIT IF DONE
7698  032552  000137  032004                  JMP     10$                     ;GO DO NEXT SECTOR
7699  032556                          180$:
7700  032556  000465                          BR      240$                    ;JUMP OVER STORAGE
7701  032560                          190$:
7702  032560  000000                          .WORD                           ;STORAGE FOR BIT POSITION
7703  032562                          200$:
7704                                  ;       MOV     #0,RMDC0                ;REFORMAT THE SECTOR IN 16 BIT MODE
7705                                  ;       MOV     #0,RMDA0                ;SECTOR 0,TRACK 0, CYL 0
7706  032562  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT MODE
7707  032570  012737  177776  001402          MOV     #-2,RMWC0               ;2 HEADER WORDS ONLY
```

```
7708  032576  012737  103564  001404          MOV     #BUFONE,RMBAO           ;BUFFER ADDRESS
7709  032604  012737  000062  001400          MOV     #WH,RMCS1O              ;WRITE HEAD AND DATA COMMAND
7710  032612  004737  036620                  JSR     PC,GENBUF               ;SET UP THE BUFFER
7711  032616  004737  034000                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7712  032622  154130                          .WORD   154130  ;TASK DESCRIPTOR
7713  032624  000404                          BR      210$                    ;GO TO 210$ IF NO ERROR
7714  032626  000240                          NOP                     ;RETURN HERE IF ERROR
7715  032630  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7716  032632  000137  032636                  JMP     210$                    ;GO TO 210$ IF ERROR WAS FOUND
7717  032636  012737  000063  001400  210$:   MOV     #WH!GO,RMCS1O           ;SET GO BIT
7718  032644  012702  001535                  MOV     #PUTINX,R2              ;TABLE ADDRESS
7719  032650  112722  000006                  MOVB    #RMDA,(R2)+             ;
7720  032654  112722  000034                  MOVB    #RMDC,(R2)+             ;
7721  032660  112722  000032                  MOVB    #RMOF,(R2)+             ;
7722  032664  112722  000004                  MOVB    #RMBA,(R2)+             ;
7723  032670  112722  000002                  MOVB    #RMWC,(R2)+             ;
7724  032674  112722  000000                  MOVB    #RMCS1,(R2)+            ;
7725  032700  112722  000200                  MOVB    #200,(R2)+              ;TERMINATOR
7726  032704  004737  037766                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7727  032710  000404                          BR      220$    ;GO TO 220$ IF NO ERROR
7728  032712  000240                          NOP                     ;RETURN HERE IF ERROR
7729  032714  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
7730  032716  000137  032732                  JMP     240$    ;GO TO 240$ IF ERROR WAS FOUND
7731  032722  000240              220$:       NOP
7732  032724  004737  040326                  JSR     PC,TIMOUT               ;WAIT FOR TIME OUT
7733  032730  000240                          NOP
7734  032732                      240$:
7735
7736                              ;PUT NEWTEST HERE
```

```
7737  032732                                  $EOSP:
7738  032732  000240                                  NOP
7739  032734  013700  001450                          MOV     TSTQUE,R0
7740  032740  062700  000002                          ADD     #2,R0
7741  032744  010037  001450                          MOV     R0,TSTQUE
7742  032750  005710                                  TST     (R0)
7743  032752  001402                                  BEQ     1$
7744  032754  000137  007124                          JMP     READY
7745  032760  012737  001452  001450  1$:             MOV     #TSTQUE+2,TSTQUE
7746                                    .SBTTL  END OF PASS ROUTINE
7747
7748                                    ;;**********************************************************************
7749                                    ;*INCREMENT THE PASS NUMBER ($PASS)
7750                                    ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
7751                                    ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
7752                                    ;*IF THERES A MONITOR GO TO IT
7753                                    ;*IF THERE ISN'T JUMP TO SHUT
7754
7755  032766                            $EOP:
7756  032766  000240                                  NOP
7757  032770  005037  001116                          CLR     $TSTNM          ;;ZERO THE TEST NUMBER
7758  032774  005037  001206                          CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
7759  033000  005237  001230                          INC     $PASS           ;;INCREMENT THE PASS NUMBER
7760  033004  042737  100000  001230                  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
7761  033012  005327                                  DEC     (PC)+           ;;LOOP?
7762  033014  000001                          $EOPCT: .WORD   1
7763  033016  003063                                  BGT     $DOAGN          ;;YES
7764  033020  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
7765  033022  000001                          $ENDCT: .WORD   1
7766  033024  033014                                  $EOPCT
7767  033026  104401  033034                          TYPE    ,65$            ;;TYPE ASCIZ STRING
7768  033032  000407                                  BR      64$             ;;GET OVER THE ASCIZ
7769                                    ;;65$:  .ASCIZ  <12><15>/END PASS #/
7770  033052                            64$:
7771  033052  013746  001230                          MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
7772                                                                           ;;TYPE PASS NUMBER
7773  033056  104405                                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
7774  033060  104401  033066                          TYPE    ,67$            ;;TYPE ASCIZ STRING
7775  033064  000421                                  BR      66$             ;;GET OVER THE ASCIZ
7776                                    ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
7777  033130                            66$:
7778  033130  013746  001126                          MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
7779                                                                           ;;TOTAL NUMBER OF ERRORS
7780  033134  104405                                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
7781  033136  104401  001217                          TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
7782  033142  005037  001126                          CLR     $ERTTL          ;;CLEAR ERROR TOTAL
7783  033146  013700  000042                  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
7784  033152  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
7785  033154  000005                                  RESET                   ;;CLEAR THE WORLD
7786  033156  004710                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
7787  033160  000240                                  NOP                     ;;SAVE ROOM
7788  033162  000240                                  NOP                     ;;FOR
7789  033164  000240                                  NOP                     ;;ACT11
7790  033166                            $DOAGN:
7791  033166  000137                                  JMP     @(PC)+          ;;RETURN
7792  033170  057372                          $RTNAD: .WORD   SHUT
```

```
 7793  033172     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
 7794          033176                          .EVEN
```

```
 7795                                   .SBTTL  SUBROUTINES
 7796
 7797                                   ;;*****************************************************************
 7798                                   .SBTTL  ERROR TYPEOUT ROUTINE
 7799
 7800                                   ;*THE ERROR TYPEOUT ROUTINE ASSEMBLES AND PRINTS INFORMATION
 7801                                   ;*REGARDING THE DETECTION OF AN ERROR AS FOLLOWS:
 7802                                   ;*
 7803                                   ;*      .UNIT NUMBER, TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER ARE
 7804                                   ;*PRINTED ON THE FIRST LINE;
 7805                                   ;*      .ERROR MESSAGE IS ASSEMBLED, FORMATTED AND PRINTED ON
 7806                                   ;*ONE OR MORE SUCCEEDING LINES;
 7807                                   ;*      .PAIRED LINES OF ERROR HEADERS AND ERROR DATA
 7808                                   ;*ARE PRINTED AFTER THE ERROR MESSAGE.
 7809
 7810  033176                           ERRTYP:
 7811  033176  104414                          SAVREG
 7812  033200  032777  020000  145746          BIT     #SW13,@SWR      ;INHIBIT TYPEOUTS??
 7813  033206  001402                          BEQ     1$              ;NO!!
 7814  033210  000137  033726                  JMP     21$             ;YES!!
 7815                                   ;TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER
 7816  033214  104401  001217           1$:     TYPE    ,$CRLF
 7817  033220  104401  033742                  TYPE    ,ERTY00         ;TYPE "UNT#"
 7818  033224  013746  001234                  MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
 7819                                                                  ;;TYPE UNIT NUMBER
 7820  033230  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 7821  033232     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
 7822  033233     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
 7823  033234  005037  033732                  CLR     TSTNMB          ;LOAD TEST NUMBER FOR
 7824  033240  013737  001226  033732          MOV     $TESTN,TSTNMB
 7825  033246  104401  033750                  TYPE    ,ERTY01         ;TYPE "TST#"
 7826  033252  013746  033732                  MOV     TSTNMB,-(SP)    ;;SAVE TSTNMB FOR TYPEOUT
 7827                                                                  ;;TYPE TEST NUMBER
 7828  033256  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 7829  033260     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
 7830  033261     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
 7831  033262  005037  033734                  CLR     ERRNMB          ;LOAD ERROR NUMBER FOR
 7832  033266  113737  001130  033734          MOVB    $ITEMB,ERRNMB   ;TYPEOUT
 7833  033274  001406                          BEQ     2$              ;SKIP IF NO ERROR CALLED
 7834  033276  104401  033760                  TYPE    ,ERTY02         ;TYPE "ERR#"
 7835  033302  013746  033734                  MOV     ERRNMB,-(SP)    ;;SAVE ERRNMB FOR TYPEOUT
 7836                                                                  ;;TYPE ERROR NUMBER
 7837  033306  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 7838  033310     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
 7839  033311     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
 7840  033312  104401  033767           2$:     TYPE    ,ERTY03         ;TYPE "PC="
 7841  033316  013746  001132                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
 7842                                                                  ;;TYPE PROGRAM COUNTER
 7843  033322  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 7844  033324     006                          .BYTE   6               ;;TYPE 6 DIGIT(S)
 7845  033325     001                          .BYTE   1               ;;TYPE LEADING ZEROS
 7846                                   ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
 7847  033326  005737  033734           3$:     TST     ERRNMB          ;WAS AN ERROR CALLED?
 7848  033332  001575                          BEQ     21$             ;NO!!
 7849  033334  104401  001217                  TYPE    ,$CRLF          ;YES-TYPE CRLF
 7850  033340  105037  033740                  CLRB    BOTFLG          ;CLEAR BOT FLAG
```

```
7851  033344  105037  033741                  CLRB    CHRCNT          ;CLEAR CHARACTER COUNTER
7852  033350  013700  033734                  MOV     ERRNMB,R0       ;R0 POINTS TO FIRST OF
7853  033354  006300                          ASL     R0              ;FOUR ENTRIES IN ERROR
7854  033356  006300                          ASL     R0              ;TABLE
7855  033360  006300                          ASL     R0
7856  033362  062700  001554                  ADD     #$ERRTB-8.,R0
7857  033366  011001                          MOV     (R0),R1         ;R1 POINTS TO ERROR MESSAGE
7858                                                                  ;TABLE
7859  033370  001507                          BEQ     13$             ;BRANCH IF NO ERROR MESSAGE
7860                                  ;TYPE THE ERROR MESSAGE
7861  033372  012102                  4$:     MOV     (R1)+,R2        ;R2=ADDRESS OF MESSAGE STRING
7862  033374  001505                          BEQ     12$             ;BRANCH IF END OF MESSAGE
7863  033376  010237  033544                  MOV     R2,11$          ;LOAD ADDRESS OF STRING
7864  033402  005037  033736                  CLR     BOTADR          ;CLEAR BOT ADDRESS
7865  033406  112203                  5$:     MOVB    (R2)+,R3        ;END OF STRING??
7866  033410  001454                          BEQ     10$             ;YES!!
7867  033412  122703  000015                  CMPB    #CR,R3          ;CARRIAGE RETURN??
7868  033416  001003                          BNE     6$              ;NO!!
7869  033420  105037  033741                  CLRB    CHRCNT          ;YES-CLEAR CHAR COUNT
7870  033424  000770                          BR      5$              ;GET NEXT CHARACTER
7871  033426  122703  000012          6$:     CMPB    #LF,R3          ;LINE FEED??
7872  033432  001765                          BEQ     5$              ;YES-GET NEXT CHARACTER
7873  033434  122703  000011                  CMPB    #HT,R3          ;HORIZONTAL TAB??
7874  033440  001007                          BNE     8$              ;NO!!
7875  033442  105237  033741          7$:     INCB    CHRCNT          ;ADJUST CHARACTER COUNT
7876  033446  132737  000007  033741          BITB    #7,CHRCNT
7877  033454  001372                          BNE     7$
7878  033456  000407                          BR      9$
7879  033460  105237  033741          8$:     INCB    CHRCNT          ;INCREMENT CHARACTER COUNT
7880  033464  122703  000040                  CMPB    #' ,R3          ;SPACE??
7881  033470  001002                          BNE     9$              ;NO!!
7882  033472  010237  033736                  MOV     R2,BOTADR       ;SAVE ADDRESS OF SPACE
7883  033476  122737  000100  033741  9$:     CMPB    #64.,CHRCNT     ;END OF LINE??
7884  033504  103340                          BHIS    5$              ;NO!!
7885  033506  013704  033736                  MOV     BOTADR,R4       ;GET ADDRESS OF LAST SPACE
7886  033512  001007                          BNE     90$             ;BRANCH IF SPACE DETECTED
7887  033514  104401  001217                  TYPE    ,$CRLF          ;TYPE CRLF
7888  033520  105037  033741                  CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
7889  033524  013702  033544                  MOV     11$,R2          ;SET UP R2 FOR TESTING
7890  033530  000726                          BR      5$
7891  033532  105044                  90$:    CLRB    -(R4)           ;REPLACE SPACE
7892  033534  112737  177777  033740          MOVB    #-1,BOTFLG      ;SET BOT FLAG
7893  033542  104401                  10$:    TYPE                    ;TYPE ERROR MESSAGE STRING
7894  033544  000000                  11$:    .WORD                   ;STRING ADDRESS GOES HERE
7895  033546  105737  033740                  TSTB    BOTFLG          ;WAS STRING TRUNCATED??
7896  033552  001707                          BEQ     4$              ;NO!!
7897  033554  104401  001217                  TYPE    ,$CRLF          ;YES-TYPE CRLF
7898  033560  105037  033740                  CLRB    BOTFLG          ;CLEAR BOT FLAG
7899  033564  105037  033741                  CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
7900  033570  013702  033736                  MOV     BOTADR,R2       ;SETUP R2 FOR TESTING
7901  033574  010237  033544                  MOV     R2,11$          ;SETUP 11$ FOR TYPING
7902  033600  112742  000040                  MOVB    #' ,-(R2)       ;RESTORE SPACE
7903  033604  105722                          TSTB    (R2)+           ;RESTORE R2
7904  033606  000677                          BR      5$              ;TYPE REST OF STRING
7905  033610                          12$:
7906                                  ;TYPE ERROR HEADER AND ERROR DATA
```

```
7907  033610                    13$:
7908  033610  016001  000002            MOV     2(R0),R1        ;R1 POINTS TO ERROR HEADER TABLE
7909  033614  001444                    BEQ     21$             ;BRANCH IF NO HEADER
7910  033616  104401  001217            TYPE    ,$CRLF          ;(ASSUME NO DATA)
7911  033622  016002  000004            MOV     4(R0),R2        ;R2 POINTS TO DATA ADDRESS TABLE
7912  033626  016003  000006            MOV     6(R0),R3        ;R3 POINTS TO FORMAT TABLE
7913  033632  012137  033642    14$:    MOV     (R1)+,15$       ;PUT HEADER ADDRESS FOR TYPE
7914  033636  001433                    BEQ     21$             ;BRANCH IF END OF HEADERS
7915                                                            ;(ASSUME END OF DATA)
7916  033640  104401                    TYPE
7917  033642  000000    15$:    .WORD   0               ;HEADER ADDRESS GOES HERE
7918  033644  104401  001217            TYPE    ,$CRLF
7919  033650  005702            .       TST     R2              ;DATA WITH HEADER??
7920  033652  001767                    BEQ     14$             ;NO!!
7921  033654  012204                    MOV     (R2)+,R4        ;R4 POINTS TO DATA ADDRESS
7922  033656  012305                    MOV     (R3)+,R5        ;R5 POINTS TO FORMAT
7923  033660  105725    16$:    TSTB    (R5)+           ;WHAT KIND OF DATA??
7924  033662  100407                    BMI     18$             ;BINARY
7925  033664  001403                    BEQ     17$             ;OCTAL
7926  033666  013446                    MOV     @(R4)+,-(SP)    ;DECIMAL
7927  033670  104405                    TYPDS
7928  033672  000405                    BR      19$
7929  033674  013446    17$:    MOV     @(R4)+,-(SP)
7930  033676  104402                    TYPOC
7931  033700  000402                    BR      19$
7932  033702  013446    18$:    MOV     @(R4)+,-(SP)
7933  033704  104406                    TYPBN
7934  033706  005714    19$:    TST     (R4)            ;MORE DATA??
7935  033710  001403                    BEQ     20$             ;NO!!
7936  033712  104401  033775            TYPE    ,ERTY04         ;YES-TYPE 2 SPACES
7937  033716  000760                    BR      16$             ;AND CONTINUE
7938  033720  104401  001217    20$:    TYPE    ,$CRLF          ;TYPE ONE BLANK LINE
7939  033724  000742                    BR      14$             ;BEFORE NEXT HEADER
7940  033726  104415    21$:    RESREG
7941  033730  000207                    RTS     PC
7942
7943  033732  000000    TSTNMB: .WORD   0               ;TEST NUMBER
7944  033734  000000    ERRNMB: .WORD   0               ;ERROR NUMBER
7945  033736  000000    BOTADR: .WORD   0               ;BEGINNING OF TEXT ADDRESS
7946  033740     000    BOTFLG: .BYTE   0               ;BOT FLAG
7947  033741     000    CHRCNT: .BYTE   0               ;CHARACTER COUNT
7948
7949  033742  047125  052111  000043    ERTY00: .ASCIZ  @UNIT#@
7950  033750  020054  042524  052123    ERTY01: .ASCIZ  @, TEST#@
7951  033756  000043
7952  033760  020054  051105  021522    ERTY02: .ASCIZ  @, ERR#@
7953  033766     000
7954  033767     054  050040  036503    ERTY03: .ASCIZ  @, PC=@
7955  033774     000
7956  033775     040  000040           ERTY04: .ASCIZ  @ @
7957                                    .EVEN
7958
```

```
                                .SBTTL  TEST PREPARATION MODULE

                                ; THIS MODULE PREPARES THE RM03 SUBSYSTEM FOR THE EXECUTION OF A TEST,
                                ; REPORTING AN ERROR TO THE USER IF AN ERROR IS  DETECTED.  THE USER
                                ; SPECIFIES TASKS TO BE PERFORMED, WHICH THE MODULE EXECUTES
                                ; USING SUBROUTINES.

                                ;CALL:
                                ;       JSR     PC,TSTPRP
                                ;       .WORD                   TASK/VERIFY DESCRIPTOR
                                ;       BR      ??              RETURN HERE IF NO ERROR
                                ;       NOP                     RETURN HERE IF ERROR
                                ;       ERROR                   ERROR DEFINED BY MODULE

                                ;TASK/VERIFY DESCRIPTOR
                                ;       BIT 15 = 1      SELECT DEVICE AND VERIFY DEVICE IS AVAILABLE
                                ;       BIT 14 = 1      CLEAR CONTROLLER AND SELECT DEVICE
                                ;       BIT 13          (RESERVED FOR DRIVE CLEAR)
                                ;       BIT 12 = 1      PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                ;       BIT 11 = 1      RECALIBRATE IF POSITIONING IN PROGRESS
                                ;       BIT 10
                                ;       BIT 9
                                ;       BIT 8
                                ;       BIT 7
                                ;       BIT 6  = 1      VERIFY CONTROLLER CLEAR OPERATION
                                ;       BIT 5           (RESERVED FOR DRIVE CLEAR)
                                ;       BIT 4  = 1      VERIFY PACK ACKNOWLEDGE
                                ;       BIT 3  = 1      VERIFY RECALIBRATION
                                ;       BIT 2
                                ;       BIT 1
                                ;       BIT 0

034000                          TSTPRP:

                                ;STORE TASK DESCRIPTOR AND CLEAR USER'S ERROR CALL
034000  017637  000000  034712          MOV     @(SP),500$      ;STORE DESCRIPTOR
034006  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
034012  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR CALL
034016  162716  000004                  SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
                                ;;*****************************************************************
                                ;SELECT DEVICE AND VERIFY DEVICE AVAILABLE IF BIT 15 SET IN TASK
034022  032737  100000  034712          BIT     #BIT15,500$     ;SELECT DEVICE??
034030  001414                          BEQ     30$             ;NO!!
034032  004737  045206                  JSR     PC,DEVSEL       ;GO SELECT DEVICE
034036  000411                          BR      30$             ;NO ERROR - CONTINUE
034040  000401                          BR      20$
034042  000000                  10$:    .WORD                   ;ERROR NUMBER FROM DEVSEL
034044  062716  000004          20$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
034050  113776  034042  000000          MOVB    10$,@(SP)
034056  000137  034700                  JMP     400$
034062                          30$:

                                ;;*****************************************************************
                                ;CLEAR CONTROLLER IF BIT 14 IS SET IN TASK
034062  032737  040000  034712          BIT     #BIT14,500$     ;CLEAR CONTROLLER??
034070  001453                          BEQ     120$            ;NO!!
```

```
8015  034072  004737  046660            JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
8016  034076  000411                    BR      60$             ;CONTINUE - NO ERROR
8017  034100  000401                    BR      50$
8018  034102  000000            40$:    .WORD                   ;ERROR NUMBER FROM CNTCLR
8019  034104  062716  000004    50$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8020  034110  113776  034102  000000    MOVB    40$,@(SP)
8021  034116  000137  034700            JMP     400$
8022  034122                    60$:
8023                              ;;*****************************************************************
8024                              ;VERIFY CONTROLLER CLEAR IF BIT6 SET IN TASK
8025  034122  032737  000100  034712    BIT     #BIT6,500$      ;VERIFY??
8026  034130  001433                    BEQ     120$            ;NO!!
8027  034132  004737  037432            JSR     PC,GETSTS       ;SETUP INDEX TABLE
8028  034136  004737  037516            JSR     PC,GET          ;GO GET STATUS
8029  034142  000411                    BR      90$             ;NO ERROR GETTING STATUS
8030  034144  000401                    BR      80$
8031  034146  000000            70$:    .WORD                   ;ERROR FROM GETTING STATUS
8032  034150  062716  000004    80$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8033  034154  113776  034146  000000    MOVB    70$,@(SP)
8034  034162  000137  034700            JMP     400$
8035  034166  004737  046776    90$:    JSR     PC,CLRSTS       ;GO VERIFY STATUS CLEAR
8036  034172  000412                    BR      120$            ;NO ERROR IN CLEAR
8037  034174  000401                    BR      110$
8038  034176  000000           100$:    .WORD                   ;ERROR IN STATUS CLEAR
8039  034200  005726           110$:    TST     (SP)+           ;STRIP RETURN ADDRESS TO
8040  034202  062716  000004            ADD     #4,(SP)         ;SUBROUTINE AND TRANSFER
8041  034206  113776  034176  000000    MOVB    100$,@(SP)      ;ERROR TO USER
8042  034214  000137  034700            JMP     400$
8043  034220                   120$:
8044
8045                              ;;*****************************************************************
8046                              ;EXECUTE PACK ACKNOWLEDGE IF BIT12 SET IN TASK AND VOLUME IS
8047                              ;NOT VALID
8048  034220  032737  010000  034712    BIT     #BIT12,500$     ;PACK ACKNOWLEDGE??
8049  034226  001511                    BEQ     240$            ;NO!!
8050  034230  112737  000012  001506    MOVB    #RMDS,GETINX    ;GET RMDS
8051  034236  112737  000200  001507    MOVB    #200,GETINX+1
8052  034244  004737  037516            JSR     PC,GET
8053  034250  000411                    BR      150$            ;NO ERROR GETTING RMDS
8054  034252  000401                    BR      140$
8055  034254  000000           130$:    .WORD
8056  034256  062716  000004   140$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8057  034262  113776  034254  000000    MOVB    130$,@(SP)
8058  034270  000137  034700            JMP     400$
8059  034274  032737  000100  001342   150$:    BIT     #VV,RMDS1       ;IS VOLUME VALID??
8060  034302  001063                    BNE     240$            ;YES!!
8061  034304  005037  001474            CLR     MEDENB          ;CLEAR MEDIA ENABLE
8062  034310  012737  000023  001400    MOV     #PAKACK!GO,RMCS10       ;LOAD PACK ACK COMMAND
8063  034316  112737  000000  001535    MOVB    #RMCS1,PUTINX   ;SETUP REGISTER INDEX TABLE
8064  034324  112737  000200  001536    MOVB    #200,PUTINX+1
8065  034332  004737  037766            JSR     PC,PUT          ;GO WRITE COMMAND
8066  034336  000410                    BR      180$            ;NO ERROR LOADING REGISTER
8067  034340  000401                    BR      170$
8068  034342  000000           160$:    .WORD                   ;ERROR FROM PUT SUB
8069  034344  062716  000004   170$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8070  034350  113776  034342  000000    MOVB    160$,@(SP)
```

```
8071  034356  000550                          BR      400$
8072  034360                          180$:
8073
8074                                  ;;*****************************************************************
8075
8076  034360  032737  000020  034712  ;VERIFY PACK ACKNOWLEDGE IF #BIT4 SET IN TASK
                                      BIT     #BIT4,500$      ;VERIFY PACK ACKNOWLEDGE??
8077  034366  001431                  BEQ     240$            ;NO!!
8078  034370  004737  037432          JSR     PC,GETSTS       ;SETUP FOR STATUS
8079  034374  004737  037516          JSR     PC,GET          ;GO GET STATUS
8080  034400  000410                  BR      210$            ;NO ERROR GETTING STATUS
8081  034402  000401                  BR      200$
8082  034404  000000          190$:   .WORD                   ;ERROR FROM GET SUB
8083  034406  062716  000004  200$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8084  034412  113776  034404  000000  MOVB    190$,@(SP)
8085  034420  000527                  BR      400$
8086  034422  004737  047656  210$:   JSR     PC,ACKSTS       ;GO CHECK ACKNOWLEDGE
8087  034426  000411                  BR      240$            ;NO ERROR
8088  034430  000401                  BR      230$
8089  034432  000000          220$:   .WORD                   ;PACK ACKNOWLEDGE ERROR
8090  034434  005726          230$:   TST     (SP)+           ;STRIP RETURN TO SUB AND
8091  034436  062716  000004          ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8092  034442  113776  034432  000000  MOVB    220$,@(SP)
8093  034450  000513                  BR      400$
8094  034452                          240$:
8095
8096                                  ;;*****************************************************************
8097                                  ;RECALIBRATE DRIVE IF BIT 11 SET IN TASK AND PIP IS ACTIVE
8098  034452  032737  004000  034712          BIT     #BIT11,500$     ;RECALIBRATE??
8099  034460  001513                  BEQ     410$            ;NO!!
8100  034462  112737  000012  001506  MOVB    #RMDS,GETINX    ;LOAD REGISTER INDEX TABLE
8101  034470  112737  000200  001507  MOVB    #200,GETINX+1
8102  034476  004737  037516          JSR     PC,GET          ;GO GET RMDS
8103  034502  000410                  BR      270$            ;NO ERROR GETTING RMDS
8104  034504  000401                  BR      260$
8105  034506  000000          250$:   .WORD                   ;ERROR FROM GET SUB
8106  034510  062716  000004  260$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8107  034514  113776  034506  000000  MOVB    250$,@(SP)
8108  034522  000466                  BR      400$
8109  034524  032737  020000  001342  270$:   BIT     #PIP,RMDSI      ;IS PIP ACTIVE??
8110  034532  001466                  BEQ     410$            ;NO!!
8111  034534  012737  000007  001400  MOV     #RECAL!GO,RMCS10;LOAD RECALIBRATE COMMAND
8112  034542  112737  000000  001535  MOVB    #RMCS1,PUTINX   ;AND REGISTER INDEX
8113  034550  112737  000200  001536  MOVB    #200,PUTINX+1
8114  034556  004737  037766          JSR     PC,PUT          ;GO ISSUE RECALIBRATE
8115  034562  000410                  BR      300$            ;NO ERROR
8116  034564  000401                  BR      290$
8117  034566  000000          280$:   .WORD                   ;ERROR IN REGISTER TRANSFER
8118  034570  062716  000004  290$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8119  034574  113776  034566  000000  MOVB    280$,@(SP)
8120  034602  000436                  BR      400$
8121  034604  004737  037432          300$:   JSR     PC,GETSTS       ;SETUP FOR STATUS
8122  034610  004737  040326          JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
8123
8124                                  ;;*****************************************************************
8125                                  ;VERIFY RECALIBRATE IF BIT 3 SET IN TASK
8126  034614  032737  000010  034712          BIT     #BIT3,500$      ;VERIFY RECALIBRATE??
```

```
 8127  034622  001432                        BEQ     410$            ;NO:!
 8128  034624  004737  037516                JSR     PC,GET          ;GO GET STATUS
 8129  034630  000410                        BR      330$            ;NO ERROR GETTING STATUS
 8130  034632  000401                        BR      320$
 8131  034634  000000              310$:     .WORD                   ;ERROR FROM GET
 8132  034636  062716  000004      320$:     ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 8133  034642  113776  034634  000000        MOVB    310$,@(SP)
 8134  034650  000413                        BR      400$
 8135  034652  004737  050452      330$:     JSR     PC,RCLSTS       ;GO CHECK RECALIBRATE
 8136  034656  000414                        BR      410$            ;NO ERROR DURING RECALIBRATE
 8137  034660  000401                        BR      350$
 8138  034662  000000              340$:     .WORD                   ;ERROR DURING RECALIBRATE
 8139  034664  005726              350$:     TST     (SP)+           ;STRIP RETURN TO SUB AND
 8140  034666  062716  000004              ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 8141  034672  113776  034662  000000        MOVB    340$,@(SP)
 8142  034700  162716  000002      400$:     SUB     #2,(SP)         ;MOVE SP BACK BEFORE ERROR
 8143  034704  000240                        NOP
 8144  034706  000240                        NOP
 8145  034710  000207              410$:     RTS     PC              ;RETURN TO USER
 8146
 8147  034712  000000              500$:     .WORD                   ;TASK/VERIFY DESCRIPTOR
```

```
8148                                    .SBTTL  BAD SECTOR MODULE
8149
8150                                    ;THE BAD SECTOR MODULE PERFORMS TWO MAJOR FUNCTIONS, I.E.,
8151                                    ;       RECOVER THE BAD SECTOR FILE FROM THE LAST TRACK, AND,
8152                                    ;       APPROVE THE USAGE OF A SECTOR BASED ON INFORMATION IN
8153                                    ;THE BAD SECTOR FILE OR ASSIGN A NEW SECTOR IF THE ONE ELECTED IS
8154                                    ;NOT AVAILABLE FOR USE.
8155                                    ;
8156                                    ;INFORMATION REQUIRED BY THE MODULE INCLUDES
8157                                    ;       .RMDCO, THE DESIRED CYLINDER,
8158                                    ;       .RMDAO, THE TRACK AND SECTOR ADDRESS,
8159                                    ;       .RMWCO, THE WORD COUNT,
8160                                    ;       .RMCS1O, THE COMMAND.
8161                                    ;
8162                                    ;MODULE CALL IS AS FOLLOWS,
8163                                    ;
8164                                    ;       JSR     PC,BADSCT
8165                                    ;       BR      ???             ;RETURN HERE IF NO ERROR
8166                                    ;       TYPE    ,MESSAGE        ;RETURN HERE IF THE BAD SECTOR FILE
8167                                    ;                               CANNOT BE RECOVERED
8168                                    ;       ERROR                   THE MODULE DEFINES THE ERROR NUMBER
8169                                    ;
8170                                    ;THE MODULE IS INTENDED TO BE CALLED PRIOR TO CALLING THE BUFFER
8171                                    ;GENERATOR SUBROUTINE, AND PRESERVES THE "PUT BUFFER" SO THAT THE
8172                                    ;BUFFER NEED ONLY BE FILLED ONCE FOR THE EXECUTION OF A FORMAT
8173                                    ;OPERATION.
8174                                    ;
8175                                    ;THE MODULE RETURNS TO THE CALLING TEST WITH THE APPROVED OR ASSIGNED
8176                                    ;SECTOR IN THE PUT BUFFER AND ALSO IN LOCATIONS "ASNDA" AND "ASNDC"
8177                                    ;SO THAT A REFERENCE IS AVAILABLE TO THE TEST OUTSIDE OF THE PUT BUFFER.
8178
8179
8180  034714                            BADSCT:
8181
8182                                    ;TEST "MEDIA ENABLE" TO DETERMINE WHETHER OR NOT THE BAD SECTOR FILES
8183                                    ;HAVE BEEN RECOVERED.
8184  034714  005737  001474                  TST     MEDENB
8185  034720  001402                        BEQ     10$
8186  034722  000137  036140                JMP     300$            ;BAD SECTOR FILE IS AVAILABLE
8187
8188                                    ;;*****************************************************************
8189                                    ;RECOVER THE BAD SECTOR FILE FROM THE LAST TRACK
8190  034726                            10$:
8191
8192                                    ;SAVE THE USER'S PUT BUFFER
8193  034726  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8194  034730  005000                        CLR     R0
8195  034732  016060  001400  103564  20$:  MOV     PUTBUF(R0),BUFFER(R0)
8196  034740  062700  000002                ADD     #2,R0           ;ADVANCE TO NEXT BUFFER POSITION
8197  034744  022700  000046                CMP     #46,R0          ;END OF BUFFER
8198  034750  103370                        BHIS    20$             ;NO !!
8199
8200                                    ;SET RETRY COUNT AND LOAD PUT BUFFER AND REGISTER INDEX TABLE
8201  034752  012737  000003  036606        MOV     #3,500$         ;RETRY COUNT
8202  034760  012737  002000  001406        MOV     #002000,RMDAO   ;STARTING SECTOR ADDRESS
8203  034766  012737  001466  001434        MOV     #822.,RMDCO     ;DESIRED CYLINDER
```

```
8204  034774  012737  177376  001402          MOV     #<^C258.+1>,RMWC0       ;WORD COUNT
8205  035002  012737  010000  001432          MOV     #FMT16,RMOF0    ;16 BIT FORMAT
8206  035010  012737  101544  001404          MOV     #MFGFIL,RMBA0   ;BUFFER ADDRESS
8207
8208  035016  012700  001535                  MOV     #PUTINX,R0      ;R0 POINTS TO REGISTER INDEX TABLE
8209  035022  112720  000006                  MOVB    #RMDA,(R0)+
8210  035026  112720  000034                  MOVB    #RMDC,(R0)+
8211  035032  112720  000002                  MOVB    #RMWC,(R0)+
8212  035036  112720  000032                  MOVB    #RMOF,(R0)+
8213  035042  112720  000004                  MOVB    #RMBA,(R0)+
8214  035046  112720  000000                  MOVB    #RMCS1,(R0)+
8215  035052  112720  000200                  MOVB    #200,(R0)+
8216  035056  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8217
8218                          ;SET GET INDEX TABLE FOR READING STATUS
8219  035060  004737  037432                  JSR     PC,GETSTS       ;SETUP GET INDEX REGISTER FOR STATUS
8220  035064                  30$:
8221
8222                          ;CLEAR THE DEVICE USING DRIVE CLEAR COMMAND
8223  035064  012737  000011  001400          MOV     #DRVCLR!GO,RMCS10       ;LOAD COMMAND IN PUT BUFFER
8224  035072  004737  037766                  JSR     PC,PUT          ;OUTPUT COMMAND
8225  035076  000411                          BR      45$             ;RETURN HERE IF NO ERROR
8226  035100  000401                          BR      40$             ;GET AROUND ERROR #
8227  035102  000000                  35$:    .WORD                   ;ERROR # GOES HERE
8228
8229  035104  062716  000006          40$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8230  035110  113776  035102  000000          MOVB    35$,@(SP)       ;MOVE ERROR NUMBER TO USER
8231  035116  000137  035616                  JMP     215$
8232  035122  004737  037516          45$:    JSR     PC,GET          ;GO GET STATUS
8233  035126  000411                          BR      60$             ;RETURN HERE IF NO ERROR
8234  035130  000401                          BR      55$             ;GET AROUND ERROR #
8235  035132  000000                  50$:    .WORD                   ;ERROR # GOES HERE
8236
8237  035134  062716  000006          55$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8238  035140  113776  035132  000000          MOVB    50$,@(SP)       ;MOVE ERROR # TO USERS ERROR CALL
8239  035146  000137  035616                  JMP     215$
8240
8241  035152  004737  052214          60$:    JSR     PC,DRVSTS       ;GO VERIFY DRIVE CLEAR COMMAND
8242  035156  000412                          BR      75$             ;RETURN HERE IF NO ERROR
8243  035160  000401                          BR      70$             ;GET AROUND ERROR
8244  035162  000000                  65$:    .WORD                   ;ERROR # GOES HERE
8245
8246  035164  005726                  70$:    TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
8247  035166  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8248  035172  113776  035162  000000          MOVB    65$,@(SP)       ;MOVE ERROR # TO USER CALL
8249  035200  000137  035616                  JMP     215$
8250
8251  035204                  75$:
8252
8253                          ;ISSUE A PACK ACKNOWLEDGE IF VOLUME VALID IS RESET
8254  035204  032737  000100  001342          BIT     #VV,RMDS1       ;IS VV RESET ??
8255  035212  001050                          BNE     120$            ;NO !!
8256  035214  012737  000023  001400          MOV     #PACACK!GO,RMCS10       ;LOAD COMMAND
8257  035222  004737  037766                  JSR     PC,PUT          ;GO PUT COMMAND TO DRIVE
8258  035226  000411                          BR      90$             ;RETURN HERE IF NO OUTPUT ERROR
8259  035230  000401                          BR      85$             ;GET AROUND ERROR #
```

```
8260  035232  000000                  80$:    .WORD                   ;ERROR # GOES HERE
8261
8262  035234  062716  000006          85$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8263  035240  113776  035232  000000          MOVB    80$,@(SP)       ;MOVE ERROR # TO ERROR CALL
8264  035246  000137  035616                  JMP     215$
8265  035252  004737  037516          90$:    JSR     PC,GET          ;GO GET STATUS FOR PACK ACK
8266  035256  000411                          BR      105$            ;RETURN HERE IF NO ERROR
8267  035260  000401                          BR      100$            ;GET AROUND ERROR #
8268  035262  000000                  95$:    .WORD                   ;ERROR # GOES HERE
8269
8270  035264  062716  000006          100$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8271  035270  113776  035262  000000          MOVB    95$,@(SP)       ;MOVE ERROR # TO CALL
8272  035276  000137  035616                  JMP     215$
8273
8274  035302  004737  047656          105$:   JSR     PC,ACKSTS       ;GO VERIFY ACKNOWLEDGE STATUS
8275  035306  000412                          BR      120$            ;RETURN HERE IF NO ERROR
8276  035310  000401                          BR      115$            ;GET AROUND ERROR #
8277  035312  000000                  110$:   .WORD                   ;ERROR # GOES HERE
8278
8279  035314  005726                  115$:   TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
8280  035316  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8281  035322  113776  035312  000000          MOVB    110$,@(SP)      ;MOVE ERROR # TO USERS ERROR CALL
8282  035330  000137  035616                  JMP     215$
8283
8284  035334                  120$:
8285
8286                          ;RECALIBRATE THE DRIVE IF PIP IS SET
8287  035334  032737  020000  001342          BIT     #PIP,RMDSI      ;IS PIP SET ??
8288  035342  001452                          BEQ     165$            ;NO !!
8289
8290  035344  012737  000007  001400          MOV     #RECAL!GO,RMCS10        ;LOAD RECALIBRATE COMMAND
8291  035352  004737  037766                  JSR     PC,PUT          ;PUT THE RECAL COMMAND
8292  035356  000411                          BR      135$            ;RETURN HERE IF NO ERROR
8293  035360  000401                          BR      130$            ;GET AROUND ERROR #
8294  035362  000000                  125$:   .WORD                   ;ERROR # GOES HERE
8295
8296  035364  062716  000006          130$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8297  035370  113776  035362  000000          MOVB    125$,@(SP)      ;MOVE ERROR # TO USERS CALL
8298  035376  000137  035616                  JMP     215$
8299  035402                  135$:
8300  035402  004737  040326                  JSR     PC,TIMOUT       ;WAIT FOR RECALIBRATE TO COMPLETE
8301  035406  004737  037516                  JSR     PC,GET          ;GO GET RECAL STATUS
8302  035412  000411                          BR      150$            ;RETURN HERE IF NO ERROR
8303  035414  000401                          BR      145$            ;GET AROUND ERROR #
8304  035416  000000                  140$:   .WORD                   ;ERROR # GOES HERE
8305
8306  035420  062716  000006          145$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8307  035424  113776  035416  000000          MOVB    140$,@(SP)      ;MOVE ERROR TO USERS CALL
8308  035432  000137  035616                  JMP     215$
8309
8310  035436  004737  050452          150$:   JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE STATUS
8311  035442  000412                          BR      165$            ;RETURN HERE IF NO ERROR
8312  035444  000401                          BR      160$            ;GET AROUND ERROR #
8313  035446  000000                  155$:   .WORD                   ;ERROR # GOES HERE
8314
8315  035450  005726                  160$:   TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
```

```
8316  035452  062716  000006            ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
8317  035456  113776  035446  000000    MOVB    155$,@(SP)       ;MOVE ERROR # TO USERS CALL
8318  035464  000137  035616            JMP     215$
8319
8320  035470                    165$:
8321
8322                              ;READ THE SECTOR IDENTIFIED BY RMDAO, INCLUDING HEADER AND DATA
8323
8324  035470  012737  000073  001400    MOV     #RH!GO,RMCS10    ;LOAD READ HEADER AND DATA COMMAND
8325  035476  004737  037766            JSR     PC,PUT           ;PUT COMMAND
8326  035502  000411                    BR      180$             ;RETURN HERE IF NO ERROR
8327  035504  000401                    BR      175$             ;GET AROUND ERROR #
8328  035506  000000            170$:   .WORD                    ;ERROR # GOES HERE
8329
8330  035510  062716  000006    175$:   ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
8331  035514  113776  035506  000000    MOVB    170$,@(SP)       ;MOVE ERROR # TO USERS ERROR CALL
8332  035522  000137  035616            JMP     215$
8333
8334  035526  004737  040326    180$:   JSR     PC,TIMOUT        ;WAIT FOR READ OPERATION TO COMPLETE
8335  035532  004737  037516            JSR     PC,GET           ;GO GET STATUS FOR READ OPERATION
8336  035536  000411                    BR      195$             ;RETURN HERE IF NO ERROR
8337  035540  000401                    BR      190$             ;GET AROUND ERROR #
8338  035542  000000            185$:   .WORD                    ;ERROR # GOES HERE
8339
8340  035544  062716  000006    190$:   ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
8341  035550  113776  035542  000000    MOVB    185$,@(SP)       ;MOVE ERROR # TO CALL
8342  035556  000137  035616            JMP     215$
8343
8344  035562  004737  053016    195$:   JSR     PC,DTASTS        ;GO VERIFY RESULTS OF READ OPERATION
8345  035566  000412                    BR      210$             ;RETURN HERE IF NO ERROR
8346  035570  000401                    BR      205$             ;GET AROUND ERROR #
8347  035572  000000            200$:   .WORD                    ;ERROR # GOES HERE
8348
8349  035574  005726    205$:   TST     (SP)+            ;STRIP RETURN ADDRESS TO SUBROUTINE
8350  035576  062716  000006            ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
8351  035602  113776  035572  000000    MOVB    200$,@(SP)       ;MOVE ERROR # TO USERS CALL
8352  035610  000137  035616            JMP     215$
8353
8354  035614  000456    210$:   BR      240$
8355
8356  035616            215$:
8357
8358                              ;AN ERROR HAS BEEN DETECTED IN TRYING TO READ THE BAD SECTOR FILE.
8359                              ;THE SECTOR WILL BE RETRIED IF POSSIBLE.
8360
8361  035616  032737  010000  001342    BIT     #MOL,RMDSI       ;IS MEDIUM ON LINE ??
8362  035624  001446                    BEQ     230$             ;YES !!
8363
8364  035626  005337  036606            DEC     500$             ;DECREMENT RETRY COUNT
8365  035632  100037                    BPL     225$             ;RETRY IF COUNT NOT NEG
8366
8367                              ;THE RETRY COUNT HAS EXPIRED - SEE IF THE ERROR IS MEDIA RELATED
8368  035634  013746  001344            MOV     RMER11,-(SP)     ;GET ER1
8369  035640  042716  100720            BIC     #DCK!HCRC!HCE!FER!ECH,(SP)
8370  035644  032737  000010  001372    BIT     #DPE,RMER21      ;WAS THER A DATA PARITY ERROR ??
8371  035652  001402                    BEQ     220$             ;NO !!
```

```
8372  035654  042716  000010            BIC     #PAR,(SP)          ;YES - CLEAR PAR
8373  035660  005726                220$: TST     (SP)+              ;ARE THERE ANY ERRORS NOT DUE TO MEDIA ?
8374  035662  001027                      BNE     230$               ;YES !!
8375  035664  013746  001372            MOV     RMER21,-(SP)       ;GET ER2
8376  035670  042726  100010            BIC     #BSE!DPE,(SP)+     ;CLEAR MEDIA ERRORS
8377  035674  001022                      BNE     230$               ;BRANCH IF NONMEDIA ERRORS DETECTED
8378
8379                                    ;THE ERRORS DETECTED WHILE TRYING TO RECOVER THE BAD SECTOR FILE ARE
8380                                    ;DUE TO THE MEDIA.  SEE IF THE BAD SECTOR FILE CAN BE RECOVERED FROM
8381                                    ;ANOTHER AREA ON THE LAST TRACK
8382
8383  035676  062737  000002  001406      ADD     #2,RMDAO           ;ADVANCE SECTOR ADDRESS
8384  035704  122737  000012  001406      CMPB    #10.,RMDAO         ;QUIT IF ALL MFG SECTORS HAVE BEEN
8385  035712  001413                      BEQ     230$               ;TRIED
8386  035714  122737  000040  001406      CMPB    #32.,RMDAO         ;QUIT IF ALL USER SECTORS HAVE
8387  035722  001407                      BEQ     230$               ;BEEN TRIED
8388  035724  012737  000003  036606      MOV     #3,500$            ;REINSTATE RETRY COUNT FOR THIS SECTOR
8389  035732  162716  000006        225$: SUB     #6,(SP)            ;MOVE SP BACK TO NO ERROR
8390  035736  000137  035064            JMP     30$                ;RETRY THE READ OPERATION
8391
8392  035742                        230$:
8393
8394                                    ;THE BAD SECTOR FILE CANNOT BE READ
8395  035742  162716  000004            SUB     #4,(SP)            ;MOVE SP TO ERROR RETURN
8396  035746  000137  036602            JMP     410$               ;GO TO MODULE EXIT
8397
8398  035752                        240$:
8399
8400                                    ;THE SECTOR WAS RECOVERED WITHOUT ERROR -
8401                                    ;       READ THE USER FILE IF THIS IS THE MFG FILE
8402                                    ;       OR ELSE DONE
8403
8404  035752  122737  000011  001406      CMPB    #9.,RMDAO          ;WAS THE USER FILE READ ??
8405  035760  103451                      BLO     260$               ;YES - READ IS COMPLETE
8406
8407                                    ;IF 144 IS IMPLEMENTED THEN READ THE USER FILE -
8408                                    ;       ELSE DUMMY THE BAD SECTOR FILE
8409  035762  032737  140000  101544      BIT     #MSE!USE,MFGFIL    ;ARE THE BAD SECTOR FLAGS OFF ??
8410  035770  001410                      BEQ     250$               ;YES - 144 IS DISABLED
8411
8412  035772  012737  002012  001406      MOV     #002012,RMDAO      ;READ THE USER FILE
8413  036000  012737  000003  036606      MOV     #3,500$            ;RELOAD THE RETRY COUNT FOR THIS SECTOR
8414  036006  000137  035064            JMP     30$                ;GO READ THE USER FILE
8415  036012                        250$:
8416
8417                                    ;DUMMY THE BAD SECTOR FILES
8418  036012  010046                      MOV     R0,-(SP)           ;;PUSH R0 ON STACK
8419  036014  010146                      MOV     R1,-(SP)           ;;PUSH R1 ON STACK
8420  036016  012701  000374              MOV     #252.,R1           ;R1 = NUMBER OF ENTRIES IN FILES
8421  036022  012700  000014              MOV     #14,R0             ;R0 = ADDRESS INDEX TO FILE  STORAGE
8422  036026  012760  177777  101544  255$: MOV     #-1,MFGFIL(R0)     ;ENTER ALL ONES IN MFG FILE
8423  036034  012760  177777  102554      MOV     #-1,USRFIL(R0)     ;ENTER ALL ONES IN USER FILE
8424  036042  062700  000002              ADD     #2,R0              ;ADVANCE ADDRESS
8425  036046  005301                      DEC     R1                 ;DECREMENT COUNT
8426  036050  001366                      BNE     255$               ;CONTINUE IF NOT DONE
8427  036052  012701  000006              MOV     #6.,R1             ;CLEAR ID, SERIAL NUMBER ETC
```

```
8428  036056  005000                          CLR     R0
8429  036060  005060  101544          256$:   CLR     MFGFIL(R0)
8430  036064  005060  102554                  CLR     USRFIL(R0)
8431  036070  062700  000002                  ADD     #2,R0
8432  036074  005301                          DEC     R1
8433  036076  001370                          BNE     256$
8434
8435  036100  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
8436  036102  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8437  036104                          260$:
8438
8439                                  ;SET MEDIA ENABLE AND RESTORE THE USERS PUT BUFFER
8440  036104  012737  177777  001474          MOV     #-1,MEDENB
8441
8442  036112  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8443  036114  005000                          CLR     R0              ;R0 IS REGISTER INDEX
8444  036116  016060  103564  001400  265$:   MOV     BUFFER(R0),PUTBUF(R0)
8445  036124  062700  000002                  ADD     #2,R0           ;ADVANCE R0
8446  036130  022700  000046                  CMP     #46,R0          ;DONE ??
8447  036134  103370                          BHIS    265$
8448
8449  036136  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8450  036140                          270$:
8451
8452  036140                          300$:
8453
8454                                  ;;*****************************************************************
8455                                  ;VERIFY THAT THE DESIRED SECTOR IS NOT IN THE MFG BAD SECTOR FILE
8456                                  ;AND NOT IN THE USERS BAD SECTOR FILE.  ASSIGN A NEW SECTOR IF THE
8457                                  ;SECTOR IS IN EITHER FILE.
8458
8459                                  ;LOAD INITIAL VARIABLES AND COMPUTE THE NUMBER OF SECTORS
8460  036140  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8461  036142  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8462  036144  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8463  036146  013737  001434  001476          MOV     RMDCO,ASNDC     ;LOAD REQUESTED CYLINDER, TRACK,
8464  036154  013737  001406  001500          MOV     RMDAO,ASNDA     ;AND SECTOR ADDRESS IN ASSIGNED STORAGE
8465  036162  005002                          CLR     R2              ;R2 = NUMBER OF SECTORS
8466  036164  013700  001402                  MOV     RMWCO,R0        ;R0 = WORD COUNT
8467  036170  005300                          DEC     R0              ;CONVERT FROM 2'S COMPLEMENT
8468  036172  005100                          COM     R0
8469  036174  012701  000400                  MOV     #256.,R1        ;R1 = NUMBER OF WORDS PER SECTOR
8470  036200  032737  000002  001400          BIT     #BIT1,RMCS10    ;IS THIS A HEADER AND DATA COMMAND ??
8471  036206  001402                          BEQ     305$            ;NO !!
8472  036210  012701  000402                  MOV     #258.,R1        ;CHANGE WORDS PER SECTOR
8473  036214  020100                  305$:   CMP     R1,R0           ;IS THERE A FULL SECTOR ??
8474  036216  101404                          BLOS    310$            ;YES !!
8475  036220  005700                          TST     R0              ;IS R0 ZERO ??
8476  036222  001405                          BEQ     315$            ;YES !!
8477  036224  005202                          INC     R2              ;INCREMENT FOR PARTIAL SECTOR
8478  036226  000403                          BR      315$
8479  036230  160100                  310$:   SUB     R1,R0           ;SUBTRACT ONE SECTOR FROM WORD COUNT
8480  036232  005202                          INC     R2              ;INCREMENT SECTOR COUNT
8481  036234  000767                          BR      305$
8482  036236  010237  036606          315$:   MOV     R2,500$         ;SAVE SECTOR COUNT
8483
```

```
8484  036242                          316$:
8485
8486                                  ;LOAD PARAMETERS AND SEARCH THE MFG AND USER BAD SECTOR FILES FOR
8487                                  ;THE ASSIGNED SECTOR AND ADJACENT SECTORS IF THE SECTOR COUNT IS
8488                                  ;MORE THAN ONE.
8489
8490  036242  012737  101560  036616          MOV     #MFGFIL+14,540$ ;MOVE THE STARTING ADDRESS OF MFG
8491                                                                  ;FILE TO BASE ADDRESS STORAGE
8492  036250  013737  001476  036612  320$:   MOV     ASNDC,520$      ;LOAD COMPARING CYLINDER ADDRESS
8493  036256  013737  001500  036614          MOV     ASNDA,530$      ;LOAD COMPARING TRACK, SECTOR ADDRESS
8494  036264  013737  036606  036610          MOV     500$,510$       ;LOAD NUMBER OF SECTORS TO CONFIRM
8495
8496                                  ;SETUP FOR A BINARY SEARCH OF THE CURRENT FILE FOR THE COMPARING
8497                                  ;CYLINDER, TRACK AND SECTOR ADDRESS
8498  036272  013700  036616          325$:   MOV     540$,R0         ;LOAD THE BASE ADDRESS IN R0
8499  036276  012701  000376                  MOV     #<<127.*4>/2>,R1        ;R1 = LENGTH OF FILE
8500  036302  060100                          ADD     R1,R0           ;START BINARY DIVIDE AT CENTER
8501  036304  021037  036612          330$:   CMP     (R0),520$       ;DOES TABLE ENTRY = COMPARING CYLINDER ?
8502  036310  001405                          BEQ     345$            ;YES !!
8503  036312  101002                          BHI     340$            ;BRANCH IF ENTRY > COMPARING CYLINDER
8504  036314                          335$:
8505
8506                                  ;FILE ENTRY IS LESS THAN COMPARING CYLINDER (OR TRACK AND SECTOR),
8507                                  ;SO ADD DISPLACEMENT TO CURRENT POSITION
8508  036314  060100                          ADD     R1,R0
8509  036316  000410                          BR      350$
8510  036320                          340$:
8511
8512                                  ;FILE ENTRY IS GREATER THAN COMPARING CYLINDER SO SUBTRACT DISPLACEMENT
8513                                  ;FROM CURRENT POSITION
8514  036320  160100                          SUB     R1,R0
8515  036322  000406                          BR      350$
8516  036324                          345$:
8517
8518                                  ;FILE ENTRY EQUALS COMPARING CYLINDER.  SEE IF THE NEXT ENTRY EQUALS
8519                                  ;THE COMPARING TRACK, AND SECTOR.
8520  036324  026037  000002  036614          CMP     2(R0),530$      ;ARE THEY EQUAL ??
8521  036332  001413                          BEQ     360$            ;YES !!
8522  036334  101371                          BHI     340$            ;BRANCH IF HIGHER
8523  036336  000766                          BR      335$            ;BRANCH IF LOWER
8524  036340                          350$:
8525
8526                                  ;THE POINTER (R0) HAS BEEN ADJUSTED.  HALVE THE DISPLACEMENT AND
8527                                  ;CONTINUE THE SEARCH IF DISPLACEMENT NOT ZERO
8528  036340  005701                          TST     R1              ;IS DISPLACEMENT ZERO ??
8529  036342  001440                          BEQ     370$            ;YES !!
8530  036344  006201                          ASR     R1              ;HALVE THE DISLPACEMENT
8531  036346  042701  000003                  BIC     #3,R1
8532  036352  020037  036616                  CMP     R0,540$         ;IS THE NEW POINTER WITHIN BOUNDS ??
8533  036356  103432                          BLO     370$            ;NO !!
8534  036360  000751                          BR      330$            ;CONTINUE SEARCH
8535  036362                          360$:
8536
8537                                  ;THE COMPARING CYLINDER, TRACK AND SECTOR IS IN THE BAD SECTOR FILE.
8538                                  ;ADVANCE THE ASSIGNED SECTOR AND START THE SEARCH ALL OVER.
8539  036362  105237  001500                  INCB    ASNDA           ;INCREMENT SECTOR
```

```
8540  036366  122737  000037  001500          CMPB    #31.,ASNDA      ;SECTOR OK ??
8541  036374  103022                          BHIS    365$            ;YES !!
8542  036376  105037  001500                  CLRB    ASNDA           ;CLEAR SECTOR AND ADVANCE TRACK
8543  036402  105237  001501                  INCB    ASNDA+1
8544  036406  122737  000004  001501          CMPB    #4,ASNDA+1      ;TRACK OK ??
8545  036414  103012                          BHIS    365$            ;YES !!
8546  036416  005037  001500                  CLR     ASNDA           ;CLEAR TRACK AND SECTOR
8547  036422  005237  001476                  INC     ASNDC           ;INCREMENT CYLINDER
8548  036426  022737  001466  001476          CMP     #822.,ASNDC     ;CYLINDER OK ??
8549  036434  103002                          BHIS    365$            ;YES !!
8550  036436  005037  001476                  CLR     ASNDC
8551  036442  000677                  365$:   BR      316$            ;REPEAT SEARCH
8552
8553  036444                          370$:
8554
8555                                  ;THE COMPARING SECTOR IS NOT IN THE BAD SECTOR FILES.  DECREMENT THE
8556                                  ;NUMBER OF SECTORS TO COMPARE AND SEARCH THE NEXT SECTOR IF THE NUMBER
8557                                  ;IS NOT ZERO.
8558
8559  036444  005337  036610                  DEC     510$            ;DECREMENT NUMBER OF SECTORS TO COMPARE
8560  036450  001432                          BEQ     380$            ;DONE IF ZERO
8561  036452  105237  036614                  INCB    530$            ;INCREMENT THE COMPARING SECTOR
8562  036456  122737  000037  036614          CMPB    #31.,530$       ;SECTOR OK ??
8563  036464  103022                          BHIS    375$            ;YES !!
8564  036466  105037  036614                  CLRB    530$            ;CLEAR SECTOR
8565  036472  105237  036615                  INCB    530$+1          ;INCREMENT TRACK
8566  036476  122737  000004  036615          CMPB    #4,530$+1       ;TRACK OK ??
8567  036504  103012                          BHIS    375$            ;YES !!
8568  036506  005037  036614                  CLR     530$            ;CLEAR SECTOR TRACK
8569  036512  005237  036612                  INC     520$            ;INCREMENT CYLINDER
8570  036516  022737  001466  036612          CMP     #822.,520$      ;CYLINDER OK ??
8571  036524  103002                          BHIS    375$            ;YES !!
8572  036526  005037  036612                  CLR     520$            ;CLEAR CYLINDER
8573  036532  000137  036272          375$:   JMP     325$            ;SEARCH NEXT SECTOR
8574
8575  036536                          380$:
8576
8577                                  ;THE ASSIGNED SECTOR (AND REQUIRED ADJACENT SECTORS) ARE NOT IN
8578                                  ;THE BAD SECTOR FILE JUST SEARCHED.  SEARCH THE USER FILE IF IT
8579                                  ;HAS NOT YET BEEN SEARCHED, ELSE ASSIGN THE SECTOR AND RETURN TO USER.
8580
8581  036536  022737  102570  036616          CMP     #USRFIL+14,540$ ;TEST BASE ADDRESS
8582  036544  001405                          BEQ     400$            ;DONE IF BASE = USER
8583  036546  012737  102570  036616          MOV     #USRFIL+14,540$ ;LOAD BASE ADDRESS
8584  036554  000137  036250          390$:   JMP     320$            ;SEARCH USER FILE
8585
8586
8587  036560                          400$:
8588
8589                                  ;ASSIGN THE SECTOR AND RETURN TO USER
8590  036560  013737  001476  001434          MOV     ASNDC,RMDCO     ;LOAD CYLINDER
8591  036566  013737  001500  001406          MOV     ASNDA,RMDAO     ;LOAD TRACK AND SECTOR
8592  036574  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
8593  036576  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
8594  036600  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8595  036602  000240                  410$:   NOP
```

```
 8596  036604  000207                      RTS     PC
 8597
 8598                                ;THE FOLLOWING ARE STORAGE LOCATIONS FOR THE MODULE
 8599
 8600  036606  000000                500$:   .WORD   ;RETRY COUNT/ NUMBER OF SECTORS REQUIRED
 8601  036610  000000                510$:   .WORD   ;NUMBER OF SECTORS TO COMPARE
 8602  036612  000000                520$:   .WORD   ;COMPARING CYLINDER
 8603  036614  000000                530$:   .WORD   ;COMPARING TRACK AND SECTOR
 8604  036616  000000                540$:   .WORD   ;BASE ADDRESS OF BAD SECTOR FILE BEING SEARCHED
```

```
8605                                    .SBTTL  BUFFER GENERATOR SUBROUTINE
8606
8607                                    ; THIS SUBROUTINE GENERATES A DATA BUFFER FOR WRITE COMMANDS.  THE
8608                                    ; BUFFER STARTS AT RMBA AND IS RMWC WORDS LONG.  THE BUFFER
8609                                    ; CONTAINS A REPETITIVE DATA PATTERN CONSISTING OF $TMP1 WORDS
8610                                    ; FROM THE DATA PATTERN TABLE, BEGINNING AT ADDRESS $TMP0.
8611                                    ; HEADER INFORMATION FOR THE BUFFER IS EXTRACTED FROM RMDC,
8612                                    ; RMDA AND RMOF.
8613
8614                                    ;CALL:
8615                                    ;(1)    JSR     PC,GENBUF
8616                                    ;(2)    ??                              RETURN HERE
8617
8618  036620                            GENBUF:
8619  036620  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8620  036622  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8621  036624  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8622  036626  010346                           MOV     R3,-(SP)        ;;PUSH R3 ON STACK
8623  036630  010446                           MOV     R4,-(SP)        ;;PUSH R4 ON STACK
8624  036632  013700  001404                   MOV     RMBA0,R0        ;LOAD DATA BUFFER ADDRESS
8625  036636  013701  001402                   MOV     RMWC0,R1        ;LOAD WORD COUNT
8626  036642  013737  001434  037060           MOV     RMDC0,60$       ;LOAD STARTING CYLINDER ADDRESS
8627  036650  013737  001406  037062           MOV     RMDA0,65$       ;LOAD STARTING TRACK,SECTOR ADDRESS
8628  036656  032737  000002  001400  10$:     BIT     #BIT1,RMCS10    ;WRITE HEADER & DATA??
8629  036664  001450                           BEQ     25$             ;NO!!
8630  036666  013710  037060                   MOV     60$,(R0)        ;WRITE HEADER WORD #1
8631  036672  042710  176000                   BIC     #^CCYLMSK,(R0)
8632  036676  052710  140000                   BIS     #MSE!USE,(R0)   ;SET (DISABLE) BAD SECTOR FLAGS
8633  036702  012702  000035                   MOV     #29.,R2         ;R2 = MAXIMUM SECTOR ADDRESS
8634  036706  032737  010000  001432           BIT     #FMT16,RMOF0    ;16 BIT FORMAT??
8635  036714  001404                           BEQ     15$             ;NO!!
8636  036716  052710  010000                   BIS     #FMT16,(R0)     ;SET FORMAT BIT IN HEADER
8637  036722  012702  000037                   MOV     #31.,R2         ;CHANGE MAXIMUM SECTOR ADDRESS
8638  036726  005201                    15$:    INC     R1              ;INCREMENT WORD COUNT
8639  036730  001444                           BEQ     50$             ;EXIT IF DONE
8640  036732  062700  000002                   ADD     #2,R0           ;MOVE R0 TO HEADER WORD #2
8641  036736  013720  037062                   MOV     65$,(R0)+       ;WRITE HEADER WORD #2
8642  036742  005201                           INC     R1              ;INCREMENT WORD COUNT AND
8643  036744  001436                           BEQ     50$             ;EXIT IF DONE
8644  036746  012703  037062                   MOV     #65$,R3         ;ADVANCE SECTOR ADDRESS
8645  036752  105213                           INCB    (R3)
8646  036754  120213                           CMPB    R2,(R3)         ;SECTOR OVERFLOW ??
8647  036756  103013                           BHIS    25$             ;NO !!
8648  036760  105013                           CLRB    (R3)            ;YES - CLEAR SECTOR ADDRESS
8649  036762  105263  000001                   INCB    1(R3)           ;ADVANCE TRACK ADDRESS
8650  036766  122763  000004  000001           CMPB    #4,1(R3)        ;TRACK OVERFLOW ??
8651  036774  103004                           BHIS    25$             ;NO !!
8652  036776  105063  000001                   CLRB    1(R3)           ;YES - CLEAR TRACK ADDRESS
8653  037002  105237  037060                   INCB    60$             ;ADVANCE CYLINDER ADDRESS
8654  037006  012704  000400            25$:    MOV     #256.,R4                ;LOAD SECTOR DATA COUNT
8655  037012  013702  001174            30$:    MOV     $TMP0,R2        ;LOAD PATTERN ADDRESS
8656  037016  013703  001176                    MOV     $TMP1,R3        ;LOAD PATTERN COUNT
8657  037022  012220                    40$:    MOV     (R2)+,(R0)+     ;WRITE DATA PATTERN
8658  037024  005201                           INC     R1              ;INCREMENT WORD COUNT AND
8659  037026  001405                           BEQ     50$             ;EXIT IF DONE
8660  037030  005304                           DEC     R4              ;DECREMENT SECTOR COUNT
```

```
8661  037032  001711                  BEQ     10$             ;START NEXT SECTOR IF 0
8662  037034  005303                  DEC     R3              ;DECREMENT PATTERN COUNT
8663  037036  001765                  BEQ     30$             ;RESTART PATTERN IF 0
8664  037040  000770                  BR      40$             ;CONTINUE DATA PATTERN
8665
8666  037042                  50$:
8667  037042  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
8668  037044  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
8669  037046  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
8670  037050  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
8671  037052  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
8672  037054  000240                  NOP
8673  037056  000207                  RTS     PC
8674
8675  037060  000000          60$:    .WORD                   ;CYLINDER ADDRESS STORAGE
8676  037062  000000          65$:    .WORD                   ;TRACK, SECTOR ADDRESS STORAGE
```

```
8677                                    .SBTTL  COMPARE BUFFER SUBROUTINE
8678
8679                                    ;THIS SUBROUTINE COMPARES THE CONTENTS OF BUFONE AND BUFTWO,
8680                                    ;ASSUMING THAT BUFONE IS THE BUFFER FROM WHICH DATA WAS WRITTEN
8681                                    ;AND BUFTWO IS THE BUFFER TO WHICH DATA WAS READ.  ERRORS IN BUFFER
8682                                    ; COMPARISON ARE REPORTED TO THE USER VIA THE USER'S ERROR CALL.
8683
8684                                    ;CALL:
8685                                    ;(1)    JSR     PC,CMPBUF
8686                                    ;(2)    .WORD   WRITE BUFFER ADDRESS
8687                                    ;       .WORD   READ BUFFER ADDRESS
8688                                    ;(3)    BR      ??              RETURN HERE IF NO ERROR
8689                                    ;(4)    NOP                     RETURN HERE IF ERROR
8690                                    ;(5)    ERROR                   ERROR DEFINED BY SUBROUTINE
8691                                    ;(6)    ???
8692
8693  037064                            CMPBUF:
8694  037064  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8695  037066  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8696  037070  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8697  037072  010346                           MOV     R3,-(SP)        ;;PUSH R3 ON STACK
8698
8699  037074  005037  037430                   CLR     150$            ;CLEAR CORRECTION FLAG
8700                                    ;DETERMINE IF DATA SHOULD BE CORRECTED
8701  037100  032737  100000  001344           BIT     #DCK,RMER1I     ;WAS THERE A DATA CHECK ??
8702  037106  001465                           BEQ     80$             ;NO !!
8703  037110  032737  000100  001344           BIT     #ECH,RMER1I     ;IS IT A HARD ERROR ??
8704  037116  001061                           BNE     80$             ;YES !!
8705  037120  033737  004000  001362           BIT     ECI,RMOFI       ;WAS ECC CORRECTION ALLOWED ??
8706  037126  001055                           BNE     80$             ;NO !!
8707  037130  032737  010000  001362           BIT     #FMT16,RMOFI    ;IS THIS 16 BIT FORMAT ??
8708  037136  001451                           BEQ     80$             ;NO !!
8709
8710                                    ;CORRECT DATA USING ECC INFORMATION
8711  037140  013700  001404                   MOV     RMBAO,R0        ;R0 = STARTING BUFFER ADDRESS
8712  037144  013701  001374                   MOV     RMEC1I,R1       ;R1 = ECC POSITION
8713  037150  052737  100000  037430           BIS     #BIT15,150$     ;SET CORRECTION FLAG
8714                                    ;MOVE R0 TO WORD BOUNDARY OF ERROR BURST
8715
8716  037156  022701  000020            10$:    CMP     #16.,R1         ;IS BIT POSITION > 1 WORD
8717  037162  103005                           BHIS    20$             ;NO !!
8718  037164  162701  000020                   SUB     #16.,R1         ;SUBTRACT 1 WORDS WORTH
8719  037170  062700  000002                   ADD     #2,R0           ;ADVANCE BUFFER ADDRESS 1 WORD
8720  037174  000770                           BR      10$
8721  037176  012702  000001            20$:    MOV     #1,R2           ;R2 = BIT POINTER
8722  037202  010203                           MOV     R2,R3           ;R3 = BIT NUMBER
8723                                    ;MOVE R2 TO STARTING BIT OF ERROR BURST
8724
8725  037204  020301                    30$:    CMP     R3,R1           ;IS R3 SAME AS R1 ??
8726  037206  001403                           BEQ     35$             ;YES !!
8727  037210  006302                           ASL     R2              ;SHIFT BIT POINTER
8728  037212  005203                           INC     R3              ;INCREMENT BIT NUMBER
8729  037214  000773                           BR      30$
8730  037216  012703  000013            35$:    MOV     #11.,R3         ;R3 = LENGTH OF ERROR BURST
8731
8732                                    ;CORRECT THE ERROR BURST
```

```
8733  037222  030237  001376          40$:    BIT     R2,RMEC2I       ;IS THIS BIT SET IN ECC PATTERN ??
8734  037226  001405                          BEQ     60$             ;NO - DO NOT CORRECT THIS BIT
8735  037230  030210                          BIT     R2,(R0)         ;IS THE BIT PRESENTLY SET ??
8736  037232  001402                          BEQ     50$             ;NO
8737  037234  040210                          BIC     R2,(R0)         ;RESET THE BIT
8738  037236  000401                          BR      60$
8739  037240  050210                  50$:    BIS     R2,(R0)         ;SET THE BIT
8740  037242  006302                  60$:    ASL     R2              ;SHIFT TO NEXT BIT
8741  037244  001004                          BNE     70$
8742  037246  012702  000001                  MOV     #1,R2           ;CONTINUE WITH FIRST BIT OF NEXT WORD
8743  037252  062700  000002                  ADD     #2,R0
8744  037256  005303                  70$:    DEC     R3              ;END OF BURST ??
8745  037260  001360                          BNE     40$             ;NO !!
8746  037262  017600  000010          80$:    MOV     @10(SP),R0              ;R0 = WRITE BUFFER
8747  037266  062766  000002  000010          ADD     #2,10(SP)               ;MOVE SP TO READ ADDRESS
8748  037274  017601  000010                  MOV     @10(SP),R1              ;R1 = READ BUFFER
8749  037300  062766  000002  000010          ADD     #2,10(SP)               ;MOVE SP TO RETURN ADDRESS
8750  037306  013702  001332                  MOV     RMWCI,R2                ;R2 = NUMBER OF WORDS TRANSFER
8751  037312  163702  001402                  SUB     RMWC0,R2
8752  037316  022021                  90$:    CMP     (R0)+,(R1)+             ;COMPARE DATA WORDS
8753  037320  001003                          BNE     100$                    ;EXIT IF NOT EQUAL
8754  037322  005302                          DEC     R2                      ;DECREMENT WORD COUNT
8755  037324  001374                          BNE     90$                     ;CONTINUE IF NOT DONE
8756  037326  000433                          BR      110$                    ;DONE COMPARE - NO ERROR
8757  037330                          100$:
8758  037330  014037  001140                  MOV     -(R0),$GDDAT            ;STORE GOOD DATA FOR TYPEOUT
8759  037334  014137  001142                  MOV     -(R1),$BDDAT            ;STORE BAD DATA FOR TYPEOUT
8760  037340  010037  001134                  MOV     R0,$GDADR               ;STORE ADDRESS OF GOOD DATA
8761  037344  010137  001136                  MOV     R1,$BDADR               ;STORE ADDRESS OF BAD DATA
8762  037350  010237  001174                  MOV     R2,$TMP0                ;STORE WORD COUNT OF ERROR
8763  037354  062766  000004  000010          ADD     #4,10(SP)               ;MOVE SP TO USER'S ERROR CALL
8764  037362  112776  000336  000010          MOVB    #336,@10(SP)            ;WRITE ERROR NUMBER IN CALL
8765
8766                          ;CHANGE ERROR NUMBER IF ECC CORRECTION FAILED
8767  037370  032737  100000  037430          BIT     #BIT15,150$     ;WAS ECC CORRECTION USED ??
8768  037376  001403                          BEQ     105$            ;NO !!
8769  037400  112776  000163  000010          MOVB    #163,@10(SP)    ;ECC CORRECTION FAILED
8770  037406  162766  000002  000010  105$:   SUB     #2,10(SP)               ;MOVE SP TO RETURN IF ERROR
8771  037414  000240                          NOP
8772  037416                          110$:
8773  037416  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
8774  037420  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
8775  037422  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
8776  037424  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8777  037426  000207                          RTS     PC                      ;RETURN TO USER
8778
8779  037430  000000                  150$:   .WORD                   ;ECC CORRECTION FLAG
```

```
8780                                    .SBTTL  GET STATUS SUBROUTINE
8781                                    ;
8782                                    ;THIS SUBROUTINE SETS UP THE "GET INDEX TABLE" AND THE "GET
8783                                    ;BUFFER" FOR READING ALL SUBSYSTEM REGISTERS VIA THE GET SUBROUTINE
8784                                    ;AND THEN RETURNS TO THE USER.
8785                                    ;
8786                                    ;CALL:  JSR     PC,GETSTS
8787                                    ;       ???                     RETURN HERE
8788
8789   037432                           GETSTS:
8790   037432  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8791   037434  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8792   037436  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8793   037440  012700  001506                   MOV     #GETINX,R0      ;R0= ADDRESS OF INDEX TABLE
8794   037444  012701  001400                   MOV     #RMEC2I+2,R1    ;R1 = ADDRESS OF GET BUFFER
8795   037450  012702  000046                   MOV     #RMEC2,R2       ;R2 = REGISTER INDE
8796   037454  110220                   2$:     MOVB    R2,(R0)+        ;WRITE REGISTER INDEX IN TABLE
8797   037456  005041                           CLR     -(R1)           ;CLEAR CORRESPONDING LOCATION
8798   037460  162702  000002           3$:     SUB     #2,R2           ;DECREMENT TO NEXT INDEX
8799   037464  100405                           BMI     4$              ;BRANCH OUT IF DONE
8800   037466  022702  000022                   CMP     #RMDB,R2        ;DONT WRITE RMDB INDEX
8801   037472  001370                           BNE     2$
8802   037474  005041                           CLR     -(R1)
8803   037476  000770                           BR      3$
8804   037500  112720  000200           4$:     MOVB    #200,(R0)+      ;WRITE TERMINATOR
8805   037504  012602                           MOV     (SP)+,R2        ;;POP STACK INTO R2
8806   037506  012601                           MOV     (SP)+,R1        ;;POP STACK INTO R1
8807   037510  012600                           MOV     (SP)+,R0        ;;POP STACK INTO R0
8808   037512  000240                           NOP
8809   037514  000207                           RTS     PC
8810
```

```
8811                                    .SBTTL  GET SUBROUTINE
8812
8813                                    ;THIS SUBROUTINE READS THE REGISTERS WHICH ARE LISTED IN THE
8814                                    ;"GET INDEX TABLE" AND STORES THEIR VALUES IN THE CORRESPONDING
8815                                    ;LOCATION IN THE "GET REGISTER BUFFER".  FOR EXAMPLE, AN
8816                                    ;ENTRY OF 04 IN THE TABLE WILL CAUSE THE SUBROUTINE TO
8817                                    ;READ "RMBA" AND STORE ITS CONTENTS AT THE LOCATION IN
8818                                    ;THE BUFFER ASSIGNED TO THAT REGISTER.  THE NUMBER OF
8819                                    ;REGISTERS TO BE READ IS VARIABLE FROM 1 TO 22; THE INDEX
8820                                    ;TABLE MUST BE TERMINATED WITH A CONTROL BYTE (200)
8821                                    ;WHICH SHOULD FOLLOW THE LAST ENTRY.
8822                                    ;
8823                                    ;SUBROUTINE CALL:
8824                                    ;(1)    "GET INDEX TABLE" HAS BEEN LOADED WITH REGISTER INDEX
8825                                    ;       VALUES AND TERMINATED WITH A CONTROL BYTE
8826                                    ;(2)    "GET INPUT BUFFER" IS AVAILABLE FOR USE.  (NOTE THAT
8827                                    ;       UNUSED LOCATIONS, I.E.,ENTRIES IN BUFFER CORRESPONDING
8828                                    ;       TO REGISTERS NOT READ, ARE NOT CHANGED.)
8829                                    ;(3)    JSR     PC,GET
8830                                    ;       BR      ???             RETURN HERE IF NO ERROR FOUND
8831                                    ;       NOP                     RETURN HERE IF ANY ERROR FOUND
8832                                    ;       ERROR                   SUB DEFINES ERROR NUMBER
8833                                    ;       ???
8834
8835  037516  000240                    GET:    NOP
8836  037520  062716  000004                    ADD     #4,(SP)         ;CLEAR ERROR NUMBER IN USER'S
8837  037524  105076  000000                    CLRB    @(SP)           ;ERROR CALL
8838  037530  162716  000004                    SUB     #4,(SP)
8839  037534  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8840  037536  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8841  037540  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8842  037542  010346                           MOV     R3,-(SP)        ;;PUSH R3 ON STACK
8843  037544  010446                           MOV     R4,-(SP)        ;;PUSH R4 ON STACK
8844  037546  013746  000004                   MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
8845  037552  013746  000006                   MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
8846  037556  013700  001276                   MOV     $BASE,R0
8847  037562  012702  001330                   MOV     #GETBUF,R2
8848  037566  012704  001506                   MOV     #GETINX,R4
8849  037572  012737  037700  000004          MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
8850  037600  012737  000300  000006          MOV     #PR6,ERRVEC+2
8851  037606  016037  000010  001174  1$:     MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
8852  037614  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
8853  037622  032737  004000  001176          BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
8854  037630  001007                          BNE     3$              ;YES!!
8855  037632  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN USER'S
8856  037640  112776  000112  000016          MOVB    #112,@16(SP)    ;ERROR CALL
8857  037646  000423                          BR      7$
8858  037650  105714                  3$:     TSTB    (R4)            ;DONE??
8859  037652  100433                          BMI     9$              ;YES!!
8860  037654  111401                          MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
8861  037656  042701  177700                  BIC     #^C1DXMSK,R1    ;CLEAR ANY SIGN EXTENSION
8862  037662  060001                          ADD     R0,R1
8863  037664  112403                          MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS FOR REGISTER
8864  037666  042703  177700                  BIC     #^C1DXMSK,R3    ;CLEAR ANY SIGN EXTENSION
8865  037672  060203                          ADD     R2,R3
8866  037674  011113                          MOV     (R1),(R3)       ;READ REGISTER
```

```
8867  037676  000764                          BR      3$
8868
8869  037700  022626                  5$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
8870  037702  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
8871  037710  112776  000007  000016          MOVB    #7,@16(SP)      ;USER'S ERROR CALL
8872  037716  162766  000002  000016  7$:     SUB     #2,16(SP)
8873  037724  105714                  8$:     TSTB    (R4)            ;DONE CLEARING??
8874  037726  100405                          BMI     9$              ;YES!!
8875  037730  005003                          CLR     R3              ;CLEAR REMAINING STORAGE
8876  037732  112403                          MOVB    (R4)+,R3        ;LOCATIONS
8877  037734  060203                          ADD     R2,R3
8878  037736  005013                          CLR     (R3)
8879  037740  000771                          BR      8$
8880  037742                          9$:
8881  037742  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
8882  037746  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
8883  037752  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
8884  037754  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
8885  037756  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
8886  037760  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
8887  037762  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8888  037764  000207                          RTS     PC              ;RETURN
8889
```

```
8890                                    .SBTTL  PUT SUBROUTINE
8891
8892                                    ;THIS SUBROUTINE WRITES THE REGISTERS WHICH ARE LISTED IN THE
8893                                    ;"PUT INDEX TABLE" WITH  THE CONTENTS OF THE CORRESPONDING
8894                                    ;LOCATION IN THE "PUT REGISTER BUFFER".  THE NUMBER OF
8895                                    ;REGISTERS WRITTEN IS VARIABLE; THE INDEX TABLE MUST
8896                                    ;BE TERMINATED WITH A CONTROL BYTE (200) WHICH SHOULD
8897                                    ;FOLLOW THE LAST ENTRY.
8898                                    ;
8899                                    ;SUBROUTINE CALL:
8900                                    ;
8901                                    ;(1)    "PUT INDEX TABLE" HAS BEEN LOADED WITH INDEX VALUES
8902                                    ;       OF REGISTERS TO BE WRITTEN.
8903                                    ;(2)    "PUT REGISTER BUFFER" CONTAINS CONTENTS OF EACH
8904                                    ;       REGISTER TO BE WRITTEN.
8905                                    ;(3)    JSR     PC,PUT
8906                                    ;       BR      ???             RETURN HERE IF NO ERROR FOUND
8907                                    ;       NOP                     RETURN HERE IF ANY ERROR FOUND
8908                                    ;       ERROR                   SUB DEFINES ERROR NUMBER
8909                                    ;       ???
8910
8911  037766  000240                    PUT:    NOP
8912  037770  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8913  037772  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8914  037774  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8915  037776  010346                            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
8916  040000  010446                            MOV     R4,-(SP)        ;;PUSH R4 ON STACK
8917  040002  013746  000004                    MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
8918  040006  013746  000006                    MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
8919  040012  013700  001276                    MOV     $BASE,R0
8920  040016  012702  001400                    MOV     #PUTBUF,R2
8921  040022  012704  001535                    MOV     #PUTINX,R4
8922  040026  012737  040134  000004            MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
8923  040034  012737  000300  000006            MOV     #PR6,ERRVEC+2
8924  040042  016037  000010  001174  1$:       MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
8925  040050  016037  000000  001176            MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
8926  040056  032737  004000  001176            BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
8927  040064  001007                            BNE     3$              ;YES!!
8928  040066  062766  000004  000016            ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
8929  040074  112776  000112  000016            MOVB    #112,@16(SP)    ;USER'S ERROR CALL
8930  040102  000423                            BR      7$
8931  040104  105714                    3$:     TSTB    (R4)            ;DONE??
8932  040106  100424                            BMI     9$              ;YES!!
8933  040110  111401                            MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
8934  040112  042701  177700                    BIC     #^C1DXMSK,R1    ;CLEAR ANY SIGN EXTENSION
8935  040116  060001                            ADD     R0,R1
8936  040120  112403                            MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS
8937  040122  042703  177700                    BIC     #^C1DXMSK,R3    ;CLEAR ANY SIGN EXTENSION
8938  040126  060203                            ADD     R2,R3
8939  040130  011311                            MOV     (R3),(R1)       ;WRITE REGISTER
8940  040132  000764                            BR      3$
8941
8942  040134  022626                    5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
8943  040136  062766  000004  000016            ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
8944  040144  112776  000007  000016            MOVB    #7,@16(SP)      ;USER'S ERROR CALL
8945  040152  162766  000002  000016  7$:       SUB     #2,16(SP)
```

```
8946
8947   040160                                  9$:
8948   040160  012637  000006                          MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
8949   040164  012637  000004                          MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
8950   040170  012604                                  MOV     (SP)+,R4        ;;POP STACK INTO R4
8951   040172  012603                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
8952   040174  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
8953   040176  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
8954   040200  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
8955   040202  000207                                  RTS     PC              ;RETURN
8956
```

```
                        .SBTTL  SIZE CLOCK SUBROUTINE

040204                  SIZCLK:
040204  013746  000004          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
040210  013746  000006          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
040214  012737  040252  000004  MOV     #1$,ERRVEC      ;SET UP FOR BUS TIMEOUT
040222  012737  000300  000006  MOV     #PR6,ERRVEC+2
040230  012737  177546  001502  MOV     #177546,CLKADR  ;LOAD ADDRESSES FOR KW11-L
040236  012737  000100  001504  MOV     #100,CLKVCT
040244  005777  141232          TST     @CLKADR         ;TEST FOR KW11-L PRESENT
040250  000421                  BR      3$              ;YES - KW11-L IS PRESENT
040252  022626          1$:     CMP     (SP)+,(SP)+     ;RESTORE SP
040254  012737  040304  000004  MOV     #2$,ERRVEC      ;SET UP FOR BUS TIMEOUT
040262  012737  172540  001502  MOV     #172540,CLKADR  ;LOAD ADDRESSES FOR KW11-P CLOCK
040270  012737  000104  001504  MOV     #104,CLKVCT
040276  005777  141200          TST     @CLKADR         ;TEST FOR KW11-P PRESENT
040302  000404                  BR      3$              ;YES - KW11-P IS PRESENT
040304  022626          2$:     CMP     (SP)+,(SP)+     ;RESTORE SP
040306  062766  000002  000004  ADD     #2,4(SP)        ;MOVE RETURN TO ERROR
040314                  3$:
040314  012637  000006          MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
040320  012637  000004          MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
040324  000207                  RTS     PC              ;RETURN TO USER
```

```
                        .SBTTL  TIMEOUT SUBROUTINE
                        ;
                        ;THIS SUBROUTINE WAITS FOR GO TO RESET OR FOR A TIMEOUT GREATER THAN
                        ;500 MS, WHICH EVER OCCURRS FIRST, AND THEN RETURNS.
                        ;
                        ;CALL:  JSR     PC,TIMOUT
                        ;       ???                     RETURN HERE

040326                  TIMOUT:
040326  010046                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
040330  010146                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
040332  010246                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
040334  013746  000004         MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
040340  013746  000006         MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
040344  012737  040446  000004 MOV     #4$,ERRVEC      ;SETUP FOR BUS TIMEOUT - 04 TRAP
040352  012737  000300  000006 MOV     #PR6,ERRVEC+2
040360  013700  001276         MOV     $BASE,R0        ;R0=RM03 ADDRESS
040364  013701  001502         MOV     CLKADR,R1       ;R1=CLOCK ADDRESS
040370  012702  000037         MOV     #31.,R2         ;R2=NUMBER OF CLOCK CYCLES
040374  020127  172540  1$:    CMP     R1,#172540      ;KW11-P CLOCK??
040400  001003                 BNE     2$              ;NO!!
040402  012761  000001  000002 MOV     #1,2(R1)        ;SET COUNTER
040410  012711  000005  2$:    MOV     #BIT2!BIT0,(R1) ;START COUNTER

040414  016046  000000  3$:    MOV     RMCS1(R0),-(SP) ;GET STATUS
040420  042716  177576         BIC     #^C<RDY!GO>,(SP)
040424  022726  000200         CMP     #RDY,(SP)+      ;RDY=1,GO=0??
040430  001420                 BEQ     5$              ;YES!!
040432  032711  000200         BIT     #BIT7,(R1)      ;TIMER DONE??
040436  001766                 BEQ     3$              ;NO!!
040440  005302                 DEC     R2              ;DEC NUMBER OF CYCLES
040442  001354                 BNE     1$              ;CONTINUE IF NOT DONE
040444  000412                 BR      5$
040446  022626          4$:    CMP     (SP)+,(SP)+     ;ADJUST STACK
040450  062766  000004  000012 ADD     #4,12(SP)       ;MOVE SP TO USER'S CALL
040456  112776  000007  000012 MOVB    #7,@12(SP)      ;WRITE ERROR NUMBER
040464  162766  000002  000012 SUB     #2,12(SP)

040472                  5$:
040472  012637  000006         MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
040476  012637  000004         MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
040502  012602                 MOV     (SP)+,R2        ;;POP STACK INTO R2
040504  012601                 MOV     (SP)+,R1        ;;POP STACK INTO R1
040506  012600                 MOV     (SP)+,R0        ;;POP STACK INTO R0
040510  000207                 RTS     PC              ;RETURN TO USER
```

```
                        .SBTTL  PRIMARY ERROR CHECK SUBROUTINE
                        ;
                        ;THE PURPOSE OF THIS SUBROUTINE IS TO VERIFY THAT STAUTS IS VALID AND
                        ;THAT FURTHER ERROR AND STATUS CHECKING SHOULD BE PERFORMED.  THE
                        ;FOLLOWING CHECKS ARE MADE:
                        ;
                        ;       .CURRECT UNIT IS SELECTED, I.E., THE UNIT SELECT BITS OF RMCS2
                        ;(BITS 0-2) EQUAL THE UNIT BEING TESTED;
                        ;
                        ;       .SELECTED UNIT IS AVAILABLE, I.E., DVA (BIT 11 OF RMCS1) IS SET
                        ;AND NED (BIT 12 OF RMCS2) IS RESET;
                        ;
                        ;       .LAST COMMAND WAS COMPLETED, I.E., THE MASSBUS CONTROLLER IS
                        ;READY (BIT 7 OF RMCS1) AND THE GO BIT IS RESET (BIT 0 OF RMCS1) OR THE
                        ;DRIVE READY BIT (BIT 7 OF RMDS) IS SET.
                        ;       .NO PARITY ERROR OCCURRED WHEN READING REMOTE REGISTERS,
                        ;I.E., MCPE = 0
                        ;       .NO PARITY ERROR OCCURRED WHEN WRITING REMOTE REGISTERS,
                        ;I.E., PAR = 0, OR, PAR = DPE = 1
                        ;
                        ;THE SUBROUTINE ASSUMES THAT:
                        ;
                        ;       .STATUS HAS BEEN STORED IN THE REGISTER INPUT BUFFER,
                        ;IN PARTICULAR, RMCS1, RMCS2 AND RMDS HAVE BEEN STORED IN THEIR
                        ;CORRESPONDING LOCATIONS OF THE "GET" BUFFER.
                        ;
                        ;       .($UNIT) CONTAINS THE DRIVE NUMBER
                        ;
                        ;THE SUBROUTINE IS CALLED AS FOLLOWS:
                        ;
                        ;(1)    JSR     PC,PRIERR
                        ;       BR      ???             RETURN HERE IF NO ERROR
                        ;       NOP                     RETURN HERE TO REPORT AN ERROR
                        ;       ERROR                   ERROR NUMBER DEFINED BY SUB
                        ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
                        ;       ???                     RETURN HERE IF NO MORE ERRORS
                        ;
040512                  PRIERR:
                        ;
                        ;CLEAR USER'S ERROR CALL
040512  062716  000004          ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
040516  105076  000000          CLRB    @(SP)           ;CLEAR ERROR NUMBER
040522  162716  000004          SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
                        ;
                        ;REPORT AN ERROR IF THE WRONG UNIT IS SELECTED
040526  013737  001340  001142  MOV     RMCS2I,$BDDAT   ;CORRECT UNIT SELECTED??
040534  042737  177770  001142  BIC     #^CUNTMSK,$BDDAT
040542  013737  001234  001140  MOV     $UNIT,$GDDAT    ;GOOD DATA FOR TYPEOUT
040550  042737  177770  001140  BIC     #^CUNTMSK,$GDDAT
040556  123737  001140  001142  CMPB    $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
                                                        ;DRIVE NUMBERS
040564  001415                  BEQ     1$              ;YES!!
040566  062716  000004          ADD     #4,(SP)
040572  112776  000001  000000  MOVB    #1,@(SP)        ;ERROR 1
```

```
9082  040600  162716  000002                    SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
9083  040604  004736                            JSR     PC,@(SP)+        ;REPORT WRONG UNIT SELECTED
9084  040606  162716  000010                    SUB     #10,(SP)         ;RESTORE (SP)
9085  040612  000240                            NOP
9086  040614  000137  041334                    JMP     10$              ;SKIP OTHER CHECKS
9087  040620                            1$:
9088
9089                                    ;REPORT AN ERROR IF THE DEVICE IS NOT AVAILABLE OR IF
9090                                    ;THE DEVICE IS NONEXISTANT
9091  040620  032737  004000  001330              BIT     #DVA,RMCS11      ;DEVICE AVAILABLE??
9092  040626  001045                            BNE     5$               ;YES!!
9093  040630  013737  001330  001140            MOV     RMCS11,$GDDAT    ;EXPECTED STATUS
9094  040636  052737  004000  001140            BIS     #DVA,$GDDAT
9095  040644  013737  001330  001142            MOV     RMCS11,$BDDAT    ;RECEIVED STATUS
9096  040652  062716  000004                    ADD     #4,(SP)
9097  040656  112776  000002  000000            MOVB    #2,@(SP)         ;ERROR #2
9098  040664  032737  010000  001340            BIT     #NED,RMCS21      ;WAS NED SET??
9099  040672  001414                            BEQ     2$               ;NO!!
9100  040674  013737  001340  001140            MOV     RMCS21,$GDDAT    ;EXPECTED STATUS
9101  040702  013737  001340  001142            MOV     RMCS21,$BDDAT    ;RECEIVED STATUS
9102  040710  042737  010000  001140            BIC     #NED,$GDDAT
9103  040716  112776  000003  000000            MOVB    #3,@(SP)         ;YES - CHANGE ERROR NUMBER
9104  040724  162716  000002             2$:    SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
9105  040730  004736                            JSR     PC,@(SP)+        ;REPORT DEVICE NOT AVAILABLE
9106  040732  162716  000010                    SUB     #10,(SP)         ;RESTORE (SP)
9107  040736  000240                            NOP
9108  040740  000575                            BR      10$              ;SKIP OTHER CHECKS
9109  040742                            5$:
9110
9111                                    ;REPORT AN ERROR IF MASSBUS CONTROLLER IS NOT READY
9112  040742  032737  000200  001330              BIT     #RDY,RMCS11      ;CONTROLLER READY??
9113  040750  001030                            BNE     7$               ;YES!!
9114  040752  013737  001330  001140            MOV     RMCS11,$GDDAT    ;EXPECTED STATUS
9115  040760  052737  000200  001140            BIS     #RDY,$GDDAT
9116  040766  042737  160001  001140            BIC     #SC!TRE!MCPE!GO,$GDDAT
9117  040774  013737  001330  001142            MOV     RMCS11,$BDDAT    ;RECEIVED STATUS
9118  041002  062716  000004                    ADD     #4,(SP)
9119  041006  112776  000004  000000            MOVB    #4,@(SP)         ;ERROR #4
9120  041014  162716  000002                    SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
9121  041020  004736                            JSR     PC,@(SP)+        ;REPORT CONTROLLER NOT READY
9122  041022  162716  000010                    SUB     #10,(SP)         ;RESTORE (SP)
9123  041026  000240                            NOP
9124  041030  000541                            BR      10$              ;SKIP OTHER CHECKS
9125  041032                            7$:
9126
9127                                    ;REPORT AN ERROR IF GO IS NOT ZERO AND DRY IS NOT ONE
9128  041032  032737  000001  001330              BIT     #GO,RMCS11       ;GO RESET??
9129  041040  001431                            BEQ     8$               ;YES!!
9130  041042  032737  000200  001342            BIT     #DRY,RMDS1       ;DRIVE READY??
9131  041050  001025                            BNE     8$               ;YES!!
9132  041052  013737  001330  001140            MOV     RMCS11,$GDDAT    ;EXPECTED STATUS
9133  041060  042737  160001  001140            BIC     #SC!TRE!MCPE!GO,$GDDAT
9134  041066  013737  001330  001142            MOV     RMCS11,$BDDAT    ;RECEIVED STATUS
9135  041074  062716  000004                    ADD     #4,(SP)
9136  041100  112776  000005  000000            MOVB    #5,@(SP)         ;ERROR #5
9137  041106  162716  000002                    SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
```

```
9138  041112  004736                          JSR     PC,@(SP)+       ;REPORT DRIVE NOT READY
9139  041114  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
9140  041120  000240                          NOP
9141  041122  000504                          BR      10$
9142  041124                          8$:
9143
9144                                          ;REPORT AN ERROR IF THE RH CONTROLLER DETECTED BAD
9145                                          ;PARITY ON THE MASSBUS CONTROL BUS
9146  041124  032737  020000  001330          BIT     #MCPE,RMCS1I    ;PARITY ERROR ??
9147  041132  001425                          BEQ     9$              ;NO!!
9148  041134  013737  001330  001140          MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
9149  041142  042737  160001  001140          BIC     #SC!TRE!MCPE!GO,$GDDAT
9150  041150  013737  001330  001142          MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
9151  041156  062716  000004                  ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
9152  041162  112776  000013  000000          MOVB    #13,@(SP)       ;ERROR #47
9153  041170  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9154  041174  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
9155  041176  162716  000010                  SUB     #10,(SP)        ;RESTORE STACK
9156  041202  000240                          NOP
9157  041204  000453                          BR      10$
9158  041206                          9$:
9159
9160                                          ;REPORT AN ERROR IF THE RM03 DETECTED A CONTROL BUS PARITY ERROR
9161  041206  032737  000010  001344          BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
9162  041214  001451                          BEQ     11$             ;NO!!
9163  041216  032737  000010  001372          BIT     #DPE,RMER2I     ;WAS IT THE CONTROL BUS??
9164  041224  001045                          BNE     11$             ;NOT SURE!!
9165  041226  032737  000010  001414          BIT     #PAR,RMER1O     ;DID TEST SET PAR ??
9166  041234  001413                          BEQ     93$             ;NO!!
9167  041236  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
9168  041240  012700  001535                  MOV     #PUTINX,R0      ;R0 POINTS TO INDEX TABLE
9169  041244  122710  000014          91$:    CMPB    #RMER1,(R0)     ;SEARCH TABLE FOR RMER1
9170  041250  001002                          BNE     92$
9171  041252  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
9172  041254  000431                          BR      11$             ;PAR WAS SET BY TEST
9173  041256  105720                  92$:    TSTB    (R0)+           ;END OF TABLE??
9174  041260  100371                          BPL     91$             ;NO!!
9175  041262  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
9176  041264  013737  001344  001140  93$:    MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
9177  041272  042737  000010  001140          BIC     #PAR,$GDDAT
9178  041300  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
9179  041306  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
9180  041312  112776  000050  000000          MOVB    #50,@(SP)       ;WRITE THE ERROR NUMBER
9181  041320  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9182  041324  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR
9183  041326  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
9184  041332  000240                          NOP
9185  041334  062716  000010          10$:    ADD     #10,(SP)                ;RETURN TO ERROR
9186  041340  000240                  11$:    NOP                     ;RETURN TO NO ERROR
9187  041342  000207                          RTS     PC
9188
```

```
                                .SBTTL  SECONDARY ERROR CHECK SUBROUTINE
                                ;
                                ;THE ERROR CHECK SUBROUTINE PROVIDES DETECTION OF SECONDARY ERRORS
                                ;SUCH AS UNEXPECTED ERRORS AND UNEXPECTED REGISTER
                                ;CONTENTS.  THESE ERRORS ARE DEEMED
                                ;SECONDARY IN THAT THEY ARE NOT NECESSARILY ASSOCIATED WITH THE OPERATION
                                ;BEING PERFORMED.  WHEN THE SUBROUTINE IDENTIFIES SUCH AN ERROR, IT MOVES
                                ;THE ERROR NUMBER TO THE ERROR CALL IN THE TEST ROUTINE AND THEN RETURNS
                                ;TO THE TEST ROUTINE WHICH MAKES THE ERROR CALL.  AFTER THE TEST ROUTINE
                                ;MAKES THE ERROR CALL, IT RETURNS TO THE SUBROUTINE WHICH THEN LOOKS FOR
                                ;OTHER ERRORS.  WHEN ALL ERRORS HAVE BEEN REPORTED, THE SUBROUTINE
                                ;RETURNS TO THE ADDRESS FOLLOWING THE SUBROUTINE CALL.
                                ;
                                ;CALL:  JSR     PC,SECERR
                                ;       BR      ???             RETURN HERE IF NO ERROR
                                ;       NOP                     RETURN HERE TO REPORT AN ERROR
                                ;       ERROR                   ERROR NUMBER DEFINED BY SUB
                                ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
                                ;       ???                     RETURN HERE IF NO MORE ERRORS
                                ;
                                ;NOTE:  THE SUBROUTINE ASSUMES THAT REGISTERS HAVE BEEN STORED AT THE
                                ;INPUT REGISTER BUFFER.
                                ;
041344                          SECERR:
                                ;
                                ;;*****************************************************************
                                ;STORE FUNCTION CODE AND CLEAR USER'S ERROR NUMBER
041344  013737  001400  045204        MOV     RMCS10,$15$     ;STORE FUNCTION CODE
041352  042737  177701  045204        BIC     #^C<F0!F1!F2!F3!F4>,$15$
041360  062716  000004                ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
041364  105076  000000                CLRB    @(SP)           ;CLEAR ERROR NUMBER
041370  162716  000004                SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
                                ;
                                ;;*****************************************************************
                                ;CHECK SECONDARY ERRORS COMMON TO ALL COMMANDS
                                ;
                                ;REPORT ERROR IF DRIVE IS NOT READY, I.E., IF DRY = 0
041374  032737  000200  001342        BIT     #DRY,RMDS1      ;DRIVE READY??
041402  001024                        BNE     5$              ;YES!!
041404  013737  001342  001142        MOV     RMDS1,$BDDAT    ;BAD DATA FOR TYPEOUT
041412  042737  177577  001142        BIC     #^CDRY,$BDDAT
041420  012737  000200  001140        MOV     #DRY,$GDDAT     ;GOOD DATA FOR TYPEOUT
041426  062716  000004                ADD     #4,(SP)
041432  112776  000010  000000        MOVB    #10,@(SP)       ;ERROR NUMBER
041440  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
041444  004736                        JSR     PC,@(SP)+       ;REPORT NOT READY
041446  162716  000010                SUB     #10,(SP)                ;RESTORE (SP) TO ERROR N
041452  000240                        NOP
                                ;
                                ;REPORT ERROR IF GO BIT IS NOT RESET
041454  032737  000001  001330  5$:   BIT     #GO,RMCS11      ;GO BIT RESET??
041462  001423                        BEQ     10$             ;YES!!
041464  013737  001330  001142        MOV     RMCS11,$BDDAT   ;BAD DATA FOR TYPEOUT
041472  042737  177776  001142        BIC     #^CGO,$BDDAT
041500  005037  001140                CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
041504  062716  000004                ADD     #4,(SP)
```

```
 9245  041510  112776  000011  000000          MOVB    #11,@(SP)       ;ERROR NUMBER
 9246  041516  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9247  041522  004736                          JSR     PC,@(SP)+       ;REPORT DEVICE NOT AVAILABLE
 9248  041524  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
 9249  041530  000240                          NOP
 9250
 9251                                  ;REPORT ERROR IF FUNCTION CODE READ FROM DEVICE IS NOT CORRECT
 9252  041532  013737  001330  001142  10$:     MOV     RMCS1I,$BDDAT   ;IS FUNCTION CODE CORRECT??
 9253  041540  042737  177701  001142          BIC     #^C76,$BDDAT
 9254  041546  013737  045204  001140          MOV     515$,$GDDAT     ;EXPECTED FUNCTION CODE
 9255  041554  023737  001142  001140          CMP     $BDDAT,$GDDAT
 9256  041562  001413                          BEQ     15$             ;YES!!
 9257  041564  062716  000004                  ADD     #4,(SP)
 9258  041570  112776  000012  000000          MOVB    #12,@(SP)       ;ERROR NUMBER
 9259  041576  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9260  041602  004736                          JSR     PC,@(SP)+       ;REPORT WRONG FUNCTION CODE
 9261  041604  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
 9262  041610  000240                          NOP
 9263  041612                          15$:
 9264                                  ;REPORT AN ERROR IF COMPOSITE ERROR IS SET AND NO OTHER
 9265                                  ;ERRORS ARE SET, OR IF COMPOSITE ERROR IS NOT SET AND
 9266                                  ;OTHER ERRORS ARE SET
 9267  041612  005037  001140                  CLR     $GDDAT          ;EXPECT "ERR"=0
 9268  041616  005737  001344                  TST     RMER1I          ;IS RMER1 =0??
 9269  041622  001003                          BNE     20$             ;NO!!
 9270  041624  005737  001372                  TST     RMER2I          ;IS RMERZ=0??
 9271  041630  001403                          BEQ     25$             ;YES!!
 9272  041632  052737  040000  001140  20$:     BIS     #ERR,$GDDAT     ;"ERR" SOULD BE SET
 9273  041640  013737  001342  001142  25$:     MOV     RMDS1,$BDDAT
 9274  041646  042737  137777  001142          BIC     #^CERR,$BDDAT
 9275  041654  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "ERR" OK??
 9276  041662  001412                          BEQ     30$             ;YES!!
 9277  041664  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
 9278  041670  112776  000047  000000          MOVB    #47,@(SP)       ;WRITE ERROR NUMBER
 9279  041676  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9280  041702  004736                          JSR     PC,@(SP)+       ;REPORT INVALID COMP ERROR
 9281  041704  162716  000010                  SUB     #10,(SP)
 9282
 9283                                  ;REPORT AN ERROR IF "TRE" IS SET AND NONE OF THE BITS WHICH SET
 9284                                  ;TRE IS SET, OR IF TRE IS NOT SET AND ONE OR MORE BITS WHICH
 9285                                  ;SET TRE IS SET
 9286  041710  005037  001140  30$:     CLR     $GDDAT          ;EXPECT "TRE" =0
 9287  041714  013746  001340                  MOV     RMCS2I,-(SP)    ;WAS DLT, WCE, UPE, NED, NEM
 9288  041720  042726  000377                  BIC     #377,(SP)+      ;PGE, MXF OR MDPE SET
 9289  041724  001010                          BNE     35$             ;YES!!
 9290  041726  032737  040000  001342          BIT     #ERR,RMDS1      ;WAS EXCEPTION RECEIVED??
 9291  041734  001407                          BEQ     40$             ;NO!!
 9292  041736  022737  000030  045204          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
 9293  041744  103003                          BHIS    40$             ;NO!!
 9294  041746  052737  040000  001140  35$:     BIS     #TRE,$GDDAT     ;"TRE" SHOULD BE SET
 9295  041754  013737  001330  001142  40$:     MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
 9296  041762  042737  137777  001142          BIC     #^CTRE,$BDDAT
 9297  041770  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "TRE" OK??
 9298  041776  001413                          BEQ     45$             ;YES!!
 9299  042000  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
 9300  042004  112776  000014  000000          MOVB    #14,@(SP)       ;WRITE ERROR NUMBER
```

```
9301  042012  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
9302  042016  004736                    JSR     PC,@(SP)+        ;REPORT TRE ERROR
9303  042020  162716  000010            SUB     #10,(SP)         ;RESTORE (SP)
9304  042024  000240                    NOP
9305  042026                     45$:
9306
9307                                     ;;*****************************************************************
9308                                     ;USING THE FUNCTION CODE TABLE, CHECK FOR THE FOLLOWING ERRORS:
9309                                     ;       .STATUS BITS NOT SET THAT SHOULD BE SET, E.G., ATA AND ILF
9310                                     ;       .STATUS BITS SET THAT SHOULD NOT BE SET, E.G., WCE AND ECH
9311                                     ;NOTE THAT SOME ERROR BITS ARE CONDITIONAL ON THE COMMAND AND OTHER
9312                                     ;STATUS CONDITIONS, E.G., WRITE LOCK ERROR SHOULD ONLY BE SET IF
9313                                     ;WRITE LOCK IS ON AND THE COMMAND IS A WRITE.
9314
9315                                     ;GET AND STORE THE ENTRY FROM THE FUNCTION CODE TABLE
9316  042026  010046                     MOV     R0,-(SP)         ;;PUSH R0 ON STACK
9317  042030  013700  045204             MOV     515$,R0          ;R0 = FUNCTION CODE
9318  042034  016037  064672  045176     MOV     FNCDTB(R0),500$  ;STORE ENTRY
9319  042042  012600                     MOV     (SP)+,R0         ;;POP STACK INTO R0
9320
9321                                     ;REPORT AN ERROR IF AN UNEXPECTED ATTENTION OCCURRED OR IF
9322                                     ;ATA IS NOT SET AND SHOULD BE SET.
9323  042044  013737  045176  001140     MOV     500$,$GDDAT      ;GET EXPECTED ATA STATUS
9324  042052  032737  040000  001342     BIT     #ERR,RMDSI       ;IS COMPOSITE ERROR SET ??
9325  042060  001403                     BEQ     50$              ;NO !!
9326  042062  052737  100000  001140     BIS     #ATA,$GDDAT      ;EXPECT AN ATTENTION
9327  042070  042737  077777  001140 50$: BIC     #^CATA,$GDDAT    ;STRIP DONT CARES
9328  042076  013737  001342  001142     MOV     RMDSI,$BDDAT     ;GET RECEIVED ATA
9329  042104  042737  077777  001142     BIC     #^CATA,$BDDAT    ;STRIP DONT CARES
9330  042112  023737  001140  001142     CMP     $GDDAT,$BDDAT    ;IS ATA OK ??
9331  042120  001413                     BEQ     55$              ;YES !!
9332  042122  062716  000004             ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
9333  042126  112776  000006  000000     MOVB    #6,@(SP)         ;LOAD ERROR # IN CALL
9334  042134  162716  000002             SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
9335  042140  004736                     JSR     PC,@(SP)+        ;REPORT ERROR
9336  042142  162716  000010             SUB     #10,(SP)         ;RESTORE SP
9337  042146  000240                     NOP
9338  042150                     55$:
9339
9340                                     ;REPORT ERROR IF ILF IS INCORRECT, I.E., IF ILF DOES NOT COMPARE
9341                                     ;WITH FUNCTION CODE TABLE
9342  042150  013737  045176  001140     MOV     500$,$GDDAT      ;GET EXPECTED ILF
9343  042156  042737  177776  001140     BIC     #^CILF,$GDDAT    ;CLEAR ALL OTHER BITS
9344  042164  013737  001344  001142     MOV     RMER1I,$BDDAT    ;GET RECEIVED ILF
9345  042172  042737  177776  001142     BIC     #^CILF,$BDDAT    ;CLEAR ALL OTHER BITS
9346  042200  023737  001140  001142     CMP     $GDDAT,$BDDAT    ;IS ILF OK ??
9347  042206  001412                     BEQ     60$              ;YES !!
9348  042210  062716  000004             ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
9349  042214  112776  000254  000000     MOVB    #254,@(SP)       ;WRITE ERROR NUMBER IN CALL
9350  042222  162716  000002             SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
9351  042226  004736                     JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
9352  042230  162716  000010             SUB     #10,(SP)         ;MOVE SP TO NO ERROR
9353  042234  005037  001140     60$:    CLR     $GDDAT           ;CLEAR EXPECTED STATUS
9354
9355                                     ;REPORT AN ERROR IF WCE IS SET AND SHOULD NOT BE SET
9356  042240  013746  045176             MOV     500$,-(SP)       ;GET WCE STATUS ENABLE
```

```
 9357  042244  052716  137777                BIS     #^CWCE,(SP)     ;SET ALL OTHER BITS
 9358  042250  013737  001340  001142        MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
 9359  042256  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR WCE IF ENABLED
 9360  042262  001412                        BEQ     90$             ;BRANCH IF WCE OK
 9361  042264  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
 9362  042270  112776  000026  000000        MOVB    #26,@(SP)       ;WRITE ERROR NUMBER
 9363  042276  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9364  042302  004736                        JSR     PC,@(SP)+       ;REPORT ERROR
 9365  042304  162716  000010                SUB     #10,(SP)        ;RESTORE ERROR
 9366  042310                        90$:
 9367
 9368                                ;REPORT ERROR IF OPI STATUS IS SET AND SHOULD NOT BE SET
 9369  042310  013746  045176                MOV     500$,-(SP)      ;GET OPI STATUS ENABLE
 9370  042314  052716  157777                BIS     #^COPI,(SP)     ;SET ALL OTHER BITS
 9371  042320  013737  001344  001142        MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
 9372  042326  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR OPI IF ENABLED
 9373  042332  001412                        BEQ     100$            ;BRANCH IF OPI OK
 9374  042334  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
 9375  042340  112776  000164  000000        MOVB    #164,@(SP)      ;WRITE ERROR NUMBER IN CALL
 9376  042346  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9377  042352  004736                        JSR     PC,@(SP)+       ;REPORT ERROR
 9378  042354  162716  000010                SUB     #10,(SP)        ;RESTORE SP
 9379  042360                        100$:
 9380
 9381                                ;REPORT ERROR IF IVC IS SET AND IS NOT ENABLED OR IF IVC IS
 9382                                ;SET AND VV IS NOT RESET
 9383  042360  013746  045176                MOV     500$,-(SP)      ;GET IVC STATUS ENABLE
 9384  042364  032737  000100  001342        BIT     #VV,RMDSI       ;IS VV SET
 9385  042372  001402                        BEQ     105$            ;NO !!
 9386  042374  042716  010000                BIC     #IVC,(SP)       ;YES - IVC SHOULD BE 0
 9387  042400  052716  167777        105$:   BIS     #^CIVC,(SP)     ;SET ALL OTHER BITS
 9388  042404  013737  001372  001142        MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
 9389  042412  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR IVC IF ENABLED
 9390  042416  001412                        BEQ     110$            ;BRANCH IF IVC OK
 9391  042420  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9392  042424  112776  000165  000000        MOVB    #165,@(SP)      ;WRITE ERROR NUMBER IN CALL
 9393  042432  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9394  042436  004736                        JSR     PC,@(SP)+       ;REPORT ERROR
 9395  042440  162716  000010                SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
 9396  042444                        110$:
 9397
 9398                                ;BIT 11 (WLE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
 9399                                ; ALL WRITE ERRORS, I.E.,
 9400                                ;       RMER1 - WLE, WCF
 9401                                ;       RMER2 - DPE
 9402                                ;       RMCS2 - UPE.
 9403                                ;EACH OF THESE ERRORS IS CHECKED TO SEE IF AN ERROR IS SET WHEN THE
 9404                                ;WRITE ERROR ENABLE BIT IS RESET.
 9405
 9406                                ;REPORT AN ERROR IF WLE IS SET AND WRITE ERRORS ARE NOT ENABLED, OR IF
 9407                                ;THE DRIVE IS NOT WRITE PROTECTED
 9408  042444  012746  177777                MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ENABLED
 9409  042450  032737  004000  045176        BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
 9410  042456  001404                        BEQ     115$            ;NO !!
 9411  042460  032737  004000  001342        BIT     #WRL,RMDSI      ;IS THE DRIVE WRITE PROTECTED ??
 9412  042466  001002                        BNE     120$            ;YES !!
```

```
9413  042470  042716  004000            115$:   BIC     #WLE,(SP)          ;RESET WLE ENABLE
9414  042474  013737  001344  001142    120$:   MOV     RMER1I,$BDDAT      ;GET RECEIVED STATUS
9415  042502  042637  001142                    BIC     (SP)+,$BDDAT       ;CLEAR WLE IF ENABLED
9416  042506  001412                            BEQ     125$               ;BRANCH IF WLE OK
9417  042510  062716  000004                    ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
9418  042514  112776  000023  000000            MOVB    #23,@(SP)          ;WRITE ERROR NUMBER IN CALL
9419  042522  162716  000002                    SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
9420  042526  004736                            JSR     PC,@(SP)+          ;REPORT ERROR AND RETURN
9421  042530  162716  000010                    SUB     #10,(SP)           ;RESTORE SP TO NO ERROR
9422  042534                            125$:
9423
9424                                    ;REPORT ERROR IF WCF IS SET AND WRITE ERRORS ARE NOT ENABLED
9425  042534  012746  177777                    MOV     #-1,-(SP)          ;ASSUME WRITE ERRORS ENABLED
9426  042540  032737  004000  045176            BIT     #WLE,500$          ;ARE WRITE ERRORS ENABLED ??
9427  042546  001002                            BNE     130$               ;YES !!
9428  042550  042716  000040                    BIC     #WCF,(SP)          ;DISABLE WCF ERROR
9429  042554  013737  001344  001142    130$:   MOV     RMER1I,$BDDAT      ;GET RECEIVED STATUS
9430  042562  042637  001142                    BIC     (SP)+,$BDDAT       ;RESET WCF IF ENABLED
9431  042566  001412                            BEQ     135$               ;BRANCH IF WCF OK
9432  042570  062716  000004                    ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
9433  042574  112776  000025  000000            MOVB    #25,@(SP)          ;WRITE ERROR NUMBER IN CALL
9434  042602  162716  000002                    SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
9435  042606  004736                            JSR     PC,@(SP)+          ;REPORT ERROR
9436  042610  162716  000010                    SUB     #10,(SP)           ;RESTORE SP TO NO ERROR
9437  042614                            135$:
9438
9439                                    ;REPORT ERROR IF DPE IS SET AND WRITE ERRORS ARE NOT ENABLED
9440  042614  012746  177777                    MOV     #-1,-(SP)          ;ASSUME WRITE ERRORS ARE ENABLED
9441  042620  032737  004000  045176            BIT     #WLE,500$          ;ARE WRITE ERRORS ENABLED ??
9442  042626  001002                            BNE     140$               ;YES !!
9443  042630  042716  000010                    BIC     #DPE,(SP)          ;RESET DPE ENABLE
9444  042634  013737  001372  001142    140$:   MOV     RMER2I,$BDDAT      ;GET RECEIVED STATUS
9445  042642  042637  001142                    BIC     (SP)+,$BDDAT       ;RESET DPE IF ENABLED
9446  042646  001412                            BEQ     145$               ;BRANCH IF DPE OK
9447  042650  062716  000004                    ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
9448  042654  112776  000040  000000            MOVB    #40,@(SP)          ;WRITE ERROR NUMBER IN CALL
9449  042662  162716  000002                    SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
9450  042666  004736                            JSR     PC,@(SP)+          ;REPORT ERROR
9451  042670  162716  000010                    SUB     #10,(SP)           ;RESTORE SP TO NO ERROR
9452  042674                            145$:
9453
9454                                    ;REPORT AN ERROR IF UPE IS SET AND WRITE ERRORS ARE NOT ENABLED
9455  042674  012746  177777                    MOV     #-1,-(SP)          ;ASSUME WRITE ERRORS ARE ENABLED
9456  042700  032737  004000  045176            BIT     #WLE,500$          ;ARE WRITE ERRORS ENABLED ??
9457  042706  001002                            BNE     150$               ;YES !!
9458  042710  042716  020000                    BIC     #UPE,(SP)          ;DISABLE UPE ERROR
9459  042714  013737  001340  001142    150$:   MOV     RMCS2I,$BDDAT      ;GET RECEIVED STATUS
9460  042722  042637  001142                    BIC     (SP)+,$BDDAT       ;RESET UPE IF ENABLED
9461  042726  001412                            BEQ     155$               ;BRANCH IF UPE OK
9462  042730  062716  000004                    ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
9463  042734  112776  000024  000000            MOVB    #24,@(SP)          ;WRITE ERROR NUMBER IN CALL
9464  042742  162716  000002                    SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
9465  042746  004736                            JSR     PC,@(SP)+          ;REPORT ERROR AND RETURN
9466  042750  162716  000010                    SUB     #10,(SP)           ;MOVE SP TO NO ERROR
9467  042754                            155$:
9468
```

```
                                        ;REPORT AN ERROR IF IAE IS SET AND IS NOT ENABLED
042754 013746 045176                            MOV     500$,-(SP)      ;GET IAE ENABLE
042760 052716 175777                            BIS     #^CIAE,(SP)     ;SET ALL OTHER BITS
042764 013737 001344 001142                     MOV     RMER11,$BDDAT   ;GET RECEIVED STATUS
042772 042637 001142                            BIC     (SP)+, $BDDAT   ;CLEAR IAE IF ENABLED
042776 001412                                   BEQ     160$            ;BRANCH IF IAE IS OK
043000 062716 000004                            ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
043004 112776 000166 000000                     MOVB    #166,@(SP)      ;WRITE ERROR NUMBER
043012 162716 000002                            SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
043016 004736                                   JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
043020 162716 000010                            SUB     #10,(SP)        ;MOVE SP TO NO ERROR
043024                                  160$:

                                        ;BIT 09 (AOE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
                                        ; ALL READ/WRITE ERRORS, I.E.,
                                        ;       RMCS1 - TRE
                                        ;       RMCS2 - DLT,NEM,MXF
                                        ;       RMDS  - LBT
                                        ;       RMER1 - AOE
                                        ;NOTE:
                                        ;       LBT IS NOT CHECKED BECAUSE IT ONLY RESETS WHEN THE DESIRED
                                        ;CYLINDER REGISTER IS WRITTEN
                                        ;NOTE:
                                        ;       AOE CANNOT BE SET IF LBT IS NOT ALSO SET
                                        ;NOTE:
                                        ;       TRE IS CHECKED AS A FUNCTION OF OTHER ERROR CONDITONS ABOVE


                                        ;REPORT AN ERROR IF DLT IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
043024 012746 177777                            MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
043030 032737 001000 045176                     BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
043036 001002                                   BNE     165$            ;YES !!
043040 042716 100000                            BIC     #DLT,(SP)       ;RESET DLT ENABLE
043044 013737 001340 001142             165$:   MOV     RMCS21,$BDDAT   ;GET RECEIVED STATUS
043052 042637 001142                            BIC     (SP)+,$BDDAT    ;CLEAR DLT IF ENABLED
043056 001412                                   BEQ     170$            ;BRANCH IF DLT IS OK
043060 062716 000004                            ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
043064 112776 000032 000000                     MOVB    #32,@(SP)       ;WRITE ERROR NUMBER IN CALL
043072 162716 000002                            SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
043076 004736                                   JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
043100 162716 000010                            SUB     #10,(SP)        ;MOVE SP TO NO ERROR
043104                                  170$:

                                        ;REPORT ERROR IF NEM IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
043104 012746 177777                            MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
043110 032737 001000 045176                     BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
043116 001002                                   BNE     175$            ;YES !!
043120 042716 004000                            BIC     #NEM,(SP)       ;DISABLE NEM
043124 013737 001340 001142             175$:   MOV     RMCS21,$BDDAT   ;GET RECEIVED STATUS
043132 042637 001142                            BIC     (SP)+,$BDDAT    ;CLEAR NEM IF ENABLED
043136 001412                                   BEQ     180$            ;BRANCH IF NEM IS OK
043140 062716 000004                            ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
043144 112776 000167 000000                     MOVB    #167,@(SP)      ;WRITE ERROR NUMBER IN CALL
043152 162716 000002                            SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
043156 004736                                   JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
043160 162716 000010                            SUB     #10,(SP)        ;MOVE SP TO NO ERROR
```

```
9525  043164                                  180$:
9526
9527                                          ;REPORT ERROR IF MXF IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
9528  043164  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
9529  043170  032737  001000  045176                  BIT     #AOE,500$       ;ARE DATA ERRORS ENABLED ??
9530  043176  001002                                  BNE     185$            ;YES !!
9531  043200  042716  001000                          BIC     #MXF,(SP)       ;DISABLE MXF ERROR
9532  043204  013737  001340  001142  185$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
9533  043212  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR MXF IF ENABLED
9534  043216  001412                                  BEQ     190$            ;BRANCH IF MXF IS OK
9535  043220  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9536  043224  112776  000033  000000                  MOVB    #33,@(SP)       ;WRITE ERROR NUMBER IN CALL
9537  043232  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9538  043236  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9539  043240  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9540  043244                                  190$:
9541
9542                                          ;REPORT ERROR IF AOE IS SET AND DATA ERRORS ARE NOT ENABLED
9543  043244  012746  177777                          MOV     #-1,-(SP)       ;ASSUME DATA ERRORS ARE ENABLED
9544  043250  032737  001000  045176                  BIT     #AOE,500$       ;ARE DATA ERRORS EANBLED ??
9545  043256  001404                                  BEQ     191$            ;NO !!
9546  043260  032737  002000  001342                  BIT     #LBT,RMDSI      ;IS LBT ALSO SET ??
9547  043266  001002                                  BNE     195$            ;YES !!
9548  043270  042716  001000                  191$:   BIC     #AOE,(SP)       ;DISABLE AOE
9549  043274  013737  001344  001142  195$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
9550  043302  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR AOE IF ENABLED
9551  043306  001412                                  BEQ     200$            ;BRANCH IF AOE IS OK
9552  043310  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9553  043314  112776  000020  000000                  MOVB    #20,@(SP)       ;WRITE ERROR NUMBER
9554  043322  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9555  043326  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9556  043330  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9557  043334                                  200$:
9558
9559                                          ;BIT 07 (HCE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
9560                                          ;HEADER ERRORS, I.E.,
9561                                          ;       RMER1 - HCRC,HCE,FER
9562                                          ;       RMER2 - BSE
9563
9564                                          ;RESET THE ENABLING BIT (HCE) IF HEADER COMPARE INHIBIT IS SET
9565  043334  032737  002000  001362                  BIT     #HCI,RMOFI      ;IS HCI SET ??
9566  043342  001403                                  BEQ     201$            ;NO !!
9567  043344  042737  000200  045176                  BIC     #HCE,500$       ;YES - DISABLE ALL HEADER ERRORS
9568  043352                                  201$:
9569
9570                                          ;REPORT AN ERROR IF HCRC IS SET AND HEADER ERRORS ARE NOT ENABLED
9571  043352  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
9572  043356  032737  000200  045176                  BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
9573  043364  001002                                  BNE     205$            ;YES !!
9574  043366  042716  000400                          BIC     #HCRC,(SP)      ;DISABLE HCRC
9575  043372  013737  001344  001142  205$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
9576  043400  042637  001142                          BIC     (SP)+,$BDDAT    ;RESET HCRC IF ENABLED
9577  043404  001412                                  BEQ     210$            ;BRANCH IF HCRC IS OK
9578  043406  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9579  043412  112776  000035  000000                  MOVB    #35,@(SP)       ;WRITE ERROR NUMBER IN CALL
9580  043420  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
```

```
9581  043424  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9582  043426  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9583  043432                          210$:
9584
9585                                  ;REPORT ERROR IF HCE IS SET AND HEADER ERRORS ARE NOT ENABLED
9586  043432  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
9587  043436  032737  000200  045176          BIT     #HCE,500$       ;ARE ERRORS ENABLED ??
9588  043444  001002                          BNE     215$            ;YES !!
9589  043446  042716  000200                  BIC     #HCE,(SP)       ;DISABLE HCE
9590  043452  013737  001344  001142  215$:   MOV     RMER11,$BDDAT   ;GET RECEIVED STATUS
9591  043460  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR HCE IF ENABLED
9592  043464  001412                          BEQ     220$            ;BRANCH IF HCE IS OK
9593  043466  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9594  043472  112776  000036  000000          MOVB    #36,@(SP)       ;WRITE ERROR NUMBER IN CALL
9595  043500  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9596  043504  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9597  043506  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9598  043512                          220$:
9599
9600                                  ;REPORT ERROR IF FER IS SET AND HEADER ERRORS ARE NOT ENABLED
9601  043512  012746  177777                  MOV     #-1,-(SP)       ;ASSUME FER IS ENABLED
9602  043516  032737  000200  045176          BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
9603  043524  001002                          BNE     225$            ;YES !!
9604  043526  042716  000020                  BIC     #FER,(SP)       ;DISABLE FER
9605  043532  013737  001344  001142  225$:   MOV     RMER11,$BDDAT   ;GET RECEIVED STATUS
9606  043540  042637  001142                  BIC     (SP)+,$BDDAT    ;RESET FER IF ENABLED
9607  043544  001412                          BEQ     230$            ;BRANCH IF FER OK
9608  043546  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9609  043552  112776  000037  000000          MOVB    #37,@(SP)       ;WRITE ERROR NUMBER IN CALL
9610  043560  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9611  043564  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9612  043566  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9613  043572                          230$:
9614
9615                                  ;REPORT ERROR IF BSE IS SET AND HEADER ERRORS ARE NOT ENABLED
9616  043572  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
9617  043576  032737  000200  045176          BIT     #HCE,500$       ;ARE THEY ENABLED ??
9618  043604  001002                          BNE     235$            ;YES !!
9619  043606  042716  100000                  BIC     #BSE,(SP)       ;DISABLE BSE
9620  043612  013737  001372  001142  235$:   MOV     RMER21,$BDDAT   ;GET RECEIVED STATUS
9621  043620  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR BSE IF ENABLED
9622  043624  001412                          BEQ     240$            ;BRANCH IF BSE OK
9623  043626  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9624  043632  112776  000354  000000          MOVB    #354,@(SP)      ;WRITE ERROR NUMBER
9625  043640  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9626  043644  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9627  043646  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9628  043652                          240$:
9629
9630                                  ;BIT 06 OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR DATA
9631                                  ;FIELD ERRORS, I.E.,
9632                                  ;       RMCS2 - MDPE
9633                                  ;       RMER1 - DCK,ECH
9634                                  ;NOTE:
9635                                  ;       ECH CANNOT SET UNLESS IT IS ENABLED AND ECI IS RESET AND
9636                                  ;DCK IS SET.
```

```
9637
9638                                        ;REPORT ERROR IF MDPE IS SET AND IS NOT ENABLED
9639  043652  012746  177777                        MOV     #-1,-(SP)       ;ASSUME ENABLED
9640  043656  032737  000100  045176                BIT     #ECH,500$       ;ARE DATA FIELD ERRORS ENABLED ??
9641  043664  001002                                BNE     245$            ;YES !!
9642  043666  042716  000400                        BIC     #MDPE,(SP)      ;DISBALE MDPE
9643  043672  013737  001340  001142  245$:         MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
9644  043700  042637  001142                        BIC     (SP)+,$BDDAT    ;CLEAR MDPE IF ENABLED
9645  043704  001412                                BEQ     250$            ;BRANCH IF MDPE OK
9646  043706  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9647  043712  112776  000027  000000                MOVB    #27,@(SP)       ;WRITE ERROR NUMBER IN CALL
9648  043720  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9649  043724  004736                                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9650  043726  162716  000010                        SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9651  043732                              250$:
9652
9653                                        ;REPORT ERROR IF DCK IS SET AND DATA FIELD ERRORS ARE NOT ENABLED
9654  043732  012746  177777                        MOV     #-1,-(SP)       ;ASSUME ENABLED
9655  043736  032737  000100  045176                BIT     #ECH,500$       ;ARE THEY ENABLED ??
9656  043744  001002                                BNE     255$            ;YES !!
9657  043746  042716  100000                        BIC     #DCK,(SP)       ;DISABLE DCK
9658  043752  013737  001344  001142  255$:         MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
9659  043760  042637  001142                        BIC     (SP)+,$BDDAT    ;CLEAR DCK IF ENABLED
9660  043764  001412                                BEQ     260$            ;BRANCH IF DCK IS OK
9661  043766  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9662  043772  112776  000030  000000                MOVB    #30,@(SP)       ;WRITE ERROR NUMBER IN CALL
9663  044000  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9664  044004  004736                                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9665  044006  162716  000010                        SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9666  044012                              260$:
9667
9668                                        ;REPORT ERROR IF ECH IS SET AND,
9669                                        ;       DATA FIELD ERRORS ARE NOT ENABLED, OR
9670                                        ;       ECI IS SET, OR
9671                                        ;       DCK IS NOT SET.
9672  044012  012746  177777                        MOV     #-1,-(SP)       ;ASSUME ENABLED
9673  044016  032737  000100  045176                BIT     #ECH,500$       ;ARE ERRORS ENABLED ??
9674  044024  001410                                BEQ     265$            ;NO !!
9675  044026  032737  004000  001362                BIT     #ECI,RMOFI      ;IS ECI SET ??
9676  044034  001004                                BNE     265$            ;YES !!
9677  044036  032737  100000  001344                BIT     #DCK,RMER1I     ;IS DCK ALSO SET ??
9678  044044  001002                                BNE     270$            ;YES !!
9679  044046  042716  000100              265$:     BIC     #ECH,(SP)       ;DISABLE ECH
9680  044052  013737  001344  001142  270$:         MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
9681  044060  042637  001142                        BIC     (SP)+,$BDDAT    ;CLEAR ECH IF ENABLED
9682  044064  001412                                BEQ     275$            ;BRANCH IF ECH IS OK
9683  044066  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9684  044072  112776  000031  000000                MOVB    #31,@(SP)       ;WRITE ERROR NUMBER IN CALL
9685  044100  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9686  044104  004736                                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9687  044106  162716  000010                        SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9688  044112                              275$:
9689
9690                                        ;;*****************************************************************
9691                                        ;PERFORM THE REMAINING ERROR CHECKS ONLY FOR DATA TRANSFER COMMANDS
9692
```

```
 9693  044112  022737  000030  045204          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
 9694  044120  103402                          BLO     280$            ;YES!!
 9695  044122  000137  045150                  JMP     355$
 9696
```

```
 9697                                        ;THE FOLLOWING STATUS CHECKS ARE FOR DATA TRANSFER COMMANDS ONLY
 9698                                        ;REPORT ERROR IF RMWC NOT ZERO AND TRE IS ZERO
 9699  044126  013737  001332  001142  280$:  MOV     RMWC1,$BDDAT    ;WORD COUNT ZERO??
 9700  044134  001421                        BEQ     285$            ;YES
 9701  044136  032737  040000  001330        BIT     #TRE,RMCS1I     ;TRANSFER ERROR DETECTED??
 9702  044144  001015                        BNE     285$            ;YES!!
 9703  044146  062716  000004                ADD     #4,(SP)
 9704  044152  112776  000015  000000        MOVB    #15,@(SP)       ;ERROR NUMBER
 9705  044160  005037  001140                CLR     $GDDAT          ;GOOD DADA FOR TYPEOUT
 9706  044164  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9707  044170  004736                        JSR     PC,@(SP)+       ;REPORT WORD COUNT NOT ZERO
 9708  044172  162716  000010                SUB     #10,(SP)        ;RESTORE (SP)
 9709  044176  000240                        NOP
 9710                                        ;REPORT ERROR IF RMBA IS NOT CORRECT
 9711  044200  013737  001332  001140  285$:  MOV     RMWC1,$GDDAT    ;NUMBER OF WORDS TRANSFERRED
 9712  044206  163737  001402  001140        SUB     RMWC0,$GDDAT
 9713  044214  006337  [illegible]            ASL     $GDDAT
 9714  044220  063737  [illegible]  001140    ADD     RMBA0,$GDDAT    ;EXPECTED BUS ADDRESS
 9715  044226  032737  [illegible]  001340    BIT     #BAI,RMCS2I     ;WAS BAI SET ??
 9716  044234  001403                        BEQ     290$            ;NO !!
 9717  044236  013737  001404  001140        MOV     RMBA0,$GDDAT    ;ADDRESS SHOULD NOT HAVE CHANGED
 9718  044244  023737  001140  001334  290$:  CMP     $GDDAT,RMBAI    ;BUS ADDRESS OK??
 9719  044252  001416                        BEQ     295$            ;YES!!
 9720  044254  013737  001334  001142        MOV     RMBA1,$BDDAT    ;BAD DATA FOR TYPEOUT
 9721  044262  062716  000004                ADD     #4,(SP)
 9722  044266  112776  000016  000000        MOVB    #16,@(SP)       ;ERROR NUMBER
 9723  044274  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9724  044300  004736                        JSR     PC,@(SP)+       ;REPORT UNEXPECTED ADDRESS
 9725  044302  162716  000010                SUB     #10,(SP)                ;RESTORE (SP)
 9726  044306  000240                        NOP
 9727                                        ;COMPUTE NUMBER OF SECTORS TRANSFERRED
 9728  044310  005046                295$:   CLR     -(SP)           ;NUMBER OF SECTORS TRANSFERRED
 9729  044312  013746  001332                MOV     RMWC1,-(SP)     ;NUMBER OF WORDS TRANSFERRED
 9730  044316  163716  001402                SUB     RMWC0,(SP)
 9731  044322  012746  000400                MOV     #256.,-(SP)     ;NUMBER OF WORDS PER SECTOR
 9732  044326  032737  000002  001400        BIT     #BIT1,RMCS1O    ;HEADER & DATA COMMAND ??
 9733  044334  001402                        BEQ     300$            ;NO !!
 9734  044336  012716  000402                MOV     #258.,(SP)      ;YES - CHANGE WORDS PER SECTOR
 9735  044342  005266  000004          300$: INC     4(SP)           ;INCREMENT SECTOR COUNT
 9736  044346  161666  000002                SUB     (SP),2(SP)      ;SUBTRACT ONE SECTOR'S WORTH
 9737  044352  003373                        BGT     300$            ;CONTINUE IF NOT DONE
 9738  044354  022626                        CMP     (SP)+,(SP)+     ;STRIP 2 FROM STACK
 9739                                        ;COMPUTE EXPECTED SECTOR, TRACK AND CYLINDER ADDRESS FROM NUMBER OF SECTORS
 9740  044356  013737  001406  045202        MOV     RMDA0,510$      ;STORE ORIGINAL SECTOR
 9741  044364  013737  001406  045200        MOV     RMDA0,505$      ;STORE ORIGINAL TRACK
 9742  044372  013737  001434  045176        MOV     RMDC0,500$      ;STORE ORIGINAL CYLINDER
 9743  044400  042737  177740  045202        BIC     #^C37,510$
 9744  044406  000337  045200                SWAB    505$
 9745  044412  042737  177770  045200        BIC     #^C7,505$
 9746  044420  062637  045202                ADD     (SP)+,510$
 9747
 9748  044424  023727  045202  000040  305$:  CMP     510$,#32.       ;SECTOR OVEFLOWED??
 9749  044432  103420                        BLO     315$            ;NO!!
 9750  044434  023727  045202  000240        CMP     510$,#<5*32.>   ;CYLINDERS WORTH??
 9751  044442  103406                        BLO     310$            ;NO!!
 9752  044444  005237  045176                INC     500$            ;YES INCREMENT CYLINDER
```

```
9753  044450  162737  000240  045202          SUB     #<5*32.>,510$   ;ADJUST SECTOR
9754  044456  000762                          BR      305$            ;TRY AGAIN
9755  044460  005237  045200          310$:   INC     505$            ;INCREMENT TRACK
9756  044464  162737  000040  045202          SUB     #32.,510$       ;ADJUST SECTOR
9757  044472  000754                          BR      305$            ;TRY AGAIN
9758
9759  044474  023727  045200  000005  315$:   CMP     505$,#5 ;TRACK OVERFLOWED??
9760  044502  103406                          BLO     320$            ;NO!!
9761  044504  005237  045176                  INC     500$            ;INCREMENT CYLINDER
9762  044510  162737  000005  045200          SUB     #5,505$ ;ADJUST TRACK
9763  044516  000766                          BR      315$            ;TRY AGAIN
9764                                  ;REPORT ERROR IF "LBT" IS NOT CORRECT
9765  044520  005037  001140          320$:   CLR     $GDDAT          ;SET GOOD DATA FOR LBT=0
9766  044524  022737  001467  045176          CMP     #823.,500$      ;SHOULD LBT BE SET??
9767  044532  101007                          BHI     325$            ;NO!!
9768  044534  032737  002000  001344          BIT     #IAE,RMER1I     ;WAS IAE SET ??
9769  044542  001003                          BNE     325$            ;YES - LBT SHOULD NOT BE SET
9770  044544  012737  002000  001140          MOV     #LBT,$GDDAT     ;SET GOOD DATA FOR LBT=1
9771  044552  013737  001342  001142  325$:   MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
9772  044560  042737  175777  001142          BIC     #^CLBT,$BDDAT
9773  044566  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS LBT CORRECT??
9774  044574  001413                          BEQ     330$            ;YES!!
9775  044576  062716  000004                  ADD     #4,(SP)
9776  044602  112776  000017  000000          MOVB    #17,@(SP)       ;ERROR NUMBER
9777  044610  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9778  044614  004736                          JSR     PC,@(SP)+       ;REPORT LBT IS WRONG
9779  044616  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
9780  044622  000240                          NOP
9781                                  ;REPORT ERROR IF "AOE" IS INCORRECT
9782  044624  005037  001140          330$:   CLR     $GDDAT          ;SET FOR AOE=0
9783  044630  032737  002000  001344          BIT     #IAE,RMER1I     ;WAS "IAE" DETECTED??
9784  044636  001031                          BNE     340$            ;YES-"AOE" SHOULD BE ZERO
9785  044640  022737  001467  045176          CMP     #823.,500$      ;SHOULD AOE BE SET??
9786  044646  101025                          BHI     340$            ;NO!!
9787  044650  005737  045200                  TST     505$            ;MAYBE
9788  044654  001012                          BNE     335$            ;YES
9789  044656  005737  045202                  TST     510$
9790  044662  001007                          BNE     335$            ;YES !!
9791  044664  032737  000010  045204          BIT     #F2,515$        ;WAS THIS READ OR WRITE CHECK ??
9792  044672  001413                          BEQ     340$            ;NO !!
9793  044674  005737  001332                  TST     RMWC1           ;WAS ALL DATA TRANSFERRED ??
9794  044700  001410                          BEQ     340$            ;YES !!
9795  044702  012737  001000  001140  335$:   MOV     #AOE,$GDDAT     ;SET FOR AOE=1
9796  044710  005037  045200                  CLR     505$            ;CLEAR EXPECTED TRACK
9797  044714  012737  000001  045202          MOV     #1,510$ ;EXPECT SECTOR=1
9798  044722  013737  001344  001142  340$:   MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
9799  044730  042737  176777  001142          BIC     #^CAOE,$BDDAT
9800  044736  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS AOE CORRECTY??
9801  044744  001413                          BEQ     345$            ;YES!!
9802  044746  062716  000004                  ADD     #4,(SP)
9803  044752  112776  000020  000000          MOVB    #20,@(SP)       ;ERROR NUMBER
9804  044760  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9805  044764  004736                          JSR     PC,@(SP)+       ;REPORT AOE IS WRONG
9806  044766  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
9807  044772  000240                          NOP
9808                                  ;REPORT ERROR IF RMDA IS NOT CORRECT
```

```
 9809  044774  032737  002000  001344  345$:   BIT     #IAE,RMER1I     ;WAS THERE AN IAE ERROR ??
 9810  045002  001062                          BNE     355$            ;YES - DONT CHECK RMDA,RMDC
 9811  045004  013737  045200  001140          MOV     505$,$GDDAT     ;SETUP EXPECTED DISK ADDRESS
 9812  045012  000337  001140                  SWAB    $GDDAT
 9813  045016  113737  045202  001140          MOVB    510$,$GDDAT
 9814  045024  013737  001336  001142          MOV     RMDAI,$BDDAT    ;SETUP RECEIVED DISK ADDRESS
 9815  045032  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
 9816  045040  001413                          BEQ     350$            ;BRANCH IF EQUAL
 9817  045042  062716  000004                  ADD     #4,(SP)
 9818  045046  112776  000021  000000          MOVB    #21,@(SP)       ;ERROR NUMBER
 9819  045054  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9820  045060  004736                          JSR     PC,@(SP)+       ;REPORT BAD DISK ADDRESS
 9821  045062  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
 9822  045066  000240                          NOP
 9823                                  ;REPORT ERROR IF RMDC IS INCORRECT
 9824  045070  013737  045176  001140  350$:   MOV     500$,$GDDAT     ;SETUP EXPECTED CYLINDER
 9825  045076  042737  176000  001140          BIC     #^C1777,$GDDAT
 9826  045104  013737  001364  001142          MOV     RMDCI,$BDDAT    ;SETUP RECEIVED CYLINDER
 9827  045112  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE CYLINDERS
 9828  045120  001413                          BEQ     355$            ;BRANCH IF EQUAL
 9829  045122  062716  000004                  ADD     #4,(SP)
 9830  045126  112776  000022  000000          MOVB    #22,@(SP)       ;ERROR NUMBER
 9831  045134  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9832  045140  004736                          JSR     PC,@(SP)+       ;REPORT BAD CYLINDER
 9833  045142  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
 9834  045146  000240                          NOP
```

```
 9835  045150  062716  000004          355$:   ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
 9836  045154  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
 9837  045160  001403                          BEQ     360$
 9838  045162  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR RETURN
 9839  045166  000402                          BR      365$
 9840  045170  162716  000004          360$:   SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
 9841  045174  000207                  365$:   RTS     PC
 9842
 9843  045176  000000                  500$:   .WORD   0               ;CYLINDER
 9844  045200  000000                  505$:   .WORD   0               ;TRACK
 9845  045202  000000                  510$:   .WORD   0               ;SECTOR
 9846  045204  000000                  515$:   .WORD   0               ;FUNCTION CODE
 9847
 9848
```

```
                                .SBTTL  DEVICE SELECT SUBROUTINE

                                ; THIS SUBROUTINE SELECTS THE DEVICE, GETTING THE DEVICE NUMBER FROM THE
                                ; TEST QUEUE.

                                ;CALL:
                                ;(1)    JSR     PC,DEVSEL
                                ;(2)    BR      ??              RETURN IF NO ERROR
                                ;(3)    NOP                     RETURN IF ERROR
                                ;(4)    ERROR                   ERROR DEFINED BY SUBROUTINE

045206                          DEVSEL:

                                ;CLEAR USER'S ERROR CALL
045206  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
045212  105076  000000                  CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF CALL
045216  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK
                                ;SAVE USER'S INFORMATION AND SETUP REGISTERS
045222  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
045226  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
045232  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
045234  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
045236  012737  045356  000004          MOV     #20$,ERRVEC     ;SETUP FOR BUS TIMEOUT
045244  012737  000300  000006          MOV     #PR6,ERRVEC+2
045252  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
045256  013701  001450                  MOV     TSTQUE,R1       ;R1 POINTS TO DEVICE NUMBER

                                ;SELECT DEVICE AND VERIFY THAT DEVICE IS AVAILABLE
045262  111160  000010                  MOVB    (R1),RMCS2(R0)  ;WRITE UNIT SELECT BITS
045266  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
045274  016037  000010  001174          MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
045302  032737  010000  001174          BIT     #NED,$TMP0      ;IS DEVICE NONEXISTENT??
045310  001407                          BEQ     10$             ;NO!!
045312  062766  000004  000010          ADD     #4,10(SP)       ;MOVE SP TO USERS ERROR CALL
045320  112776  000111  000010          MOVB    #111,@10(SP)    ;WRITE ERROR NUMBER
045326  000422                          BR      30$             ;
045330  032737  004000  001176  10$:     BIT     #DVA,$TMP1      ;IS DEVICE AVAILABLE??
045336  001021                          BNE     35$             ;YES!!
045340  062766  000004  000010          ADD     #4,10(SP)
045346  112776  000112  000010          MOVB    #112,@10(SP)
045354  000407                          BR      30$

045356                          20$:

                                ;HANDLE BUS TIMEOUT
045356  022626                          CMP     (SP)+,(SP)+     ;ADJUST SP
045360  062766  000004  000010          ADD     #4,10(SP)
045366  112776  000113  000010          MOVB    #113,@10(SP)
045374  162766  000002  000010  30$:     SUB     #2,10(SP)       ;MOVE SP TO RETURN IF ERROR

045402                          35$:

                                ;RESTORE USERS DATA AND RETURN TO ADDRESS ON STACK
045402  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
045404  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
```

```
 9905  045406  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
 9906  045412  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 9907  045416  000207                        RTS     PC              ;EXIT
```

```
                                .SBTTL  SEEK STATUS CHECK SUBROUTINE
                                ;
                                ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEEK TESTS USING STATUS
                                ;STORED IN THE GET BUFFER AND TEST PARAMETERS STORED IN THE PUT BUFFER.
                                ;
                                ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS DETECTED
                                ;AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN THE "ERROR" TRAP
                                ;OF THE CALLING ROUTINE.  SEEK STATUS IS CHECKED AS FOLLOWS:
                                ;
                                ;CALL:
                                ;(1)     JSR     PC,SEKSTS
                                ;       BR      ???             RETURN HERE IF NO ERROR
                                ;       NOP                     RETURN HERE TO REPORT AN ERROR
                                ;       ERROR                   ERROR NUMBER DEFINED BY SUB
                                ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
                                ;       ???                     RETURN HERE IF NO MORE ERRORS

045420                          SEKSTS:

                                ;CLEAR USER'S ERROR  CALL
045420  000240                          NOP
045422  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
045426  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
045432  162716  000004                  SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
045436  005037  046656                  CLR     300$            ;CLEAR ERROR FLAGS

                                ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING
                                ;LOCAL REGISTERS, I.E., "PAR"  = 1 AND "DPE" = 0
045442  032737  000010  001344          BIT     #PAR,RMER1I     ;WAS PARITY ERROR DETECTED??
045450  001424                          BEQ     1$              ;NO!!
045452  032737  000010  001372          BIT     #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
045460  001020                          BNE     1$              ;NOT SURE!!

                                ;REPORT REGISTER PARITY ERROR VIA USER'S ERROR CALL
045462  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
045466  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
045474  062716  000004                  ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
045500  112776  000050  000000          MOVB    #50,@(SP)       ;ERROR #50
045506  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
045512  004736                          JSR     PC,@(SP)+
045514  162716  000010                  SUB     #10,(SP)        ;RESTORE STACK
045520  000437                          BR      3$              ;IAE SHOULD  BE ZERO

                                ;DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK,SECTOR AND
                                ;CYLINDER ADDRESS USED DURING SEEK OPERATION
045522  012737  002000  001140  1$:     MOV     #IAE,$GDDAT     ;SET UP FOR IAE = 1
045530  052737  040000  046656          BIS     #SKI,300$       ;SET SKI ERROR FLAG
045536  023727  001434  001466          CMP     RMDC0,#822.     ;CYLINDER > 822??
045544  101025                          BHI     3$              ;YES!!
045546  042737  040000  046656          BIC     #SKI,300$       ;CLEAR SKI ERROR FLAG
045554  123727  001406  000037          CMPB    RMDA0,#31.      ;SECTOR > 31??
045562  101016                          BHI     3$              ;YES!!
045564  123727  001406  000035          CMPB    RMDA0,#29.      ;SECTOR > 29 ??
045572  101404                          BLOS    2$              ;NO!!
```

```
 9964  045574  032737  010000  001432          BIT     #FMT16,RMOFO    ;30 SECTOR FORMAT??
 9965  045602  001406                          BEQ     3$              ;YES!!
 9966  045604  123727  001407  000004  2$:     CMPB    RMDAO+1,#4.     ;TRACK  >4??
 9967  045612  101002                          BHI     3$              ;YES!!
 9968  045614  005037  001140                  CLR     $GDDAT          ;"IAE" SHOULD BE 0
 9969
 9970                                          ;COMPARE EXPECTED AND RECEIVED "IAE" STATUS
 9971  045620  013737  001344  001142  3$:     MOV     RMER1I,$BDDAT   ;IS IAE OK??
 9972  045626  042737  175777  001142          BIC     #^CIAE,$BDDAT
 9973  045634  023737  001140  001142          CMP     $GDDAT,$BDDAT
 9974  045642  001004                          BNE     35$             ;IAE IN ERROR
 9975  045644  042737  040000  046656          BIC     #SKI,300$       ;CLEAR SKI FLAG
 9976  045652  000413                          BR      5$              ;GO CHECK NEXT ERROR
 9977  045654                                  35$:
 9978                                          ;REPORT INCORRECT "IAE" STATUS VIA USER'S ERROR CALL
 9979  045654  062716  000004                  ADD     #4,(SP)
 9980  045660  112776  000051  000000          MOVB    #51,@(SP)       ;ERROR 51
 9981  045666  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9982  045672  004736                          JSR     PC,@(SP)+       ;REPORT INCORRECT IAE
 9983  045674  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
 9984  045700  000240                          NOP
 9985  045702                                  5$:
 9986
 9987                                          ;REPORT ANY IVC ERROR AS
 9988                                          ;       IVC ERROR WITH VOLUME VALID ZERO
 9989                                          ;       ERRONEOUS IVC ERROR, VOLUME VALID IS SET
 9990  045702  032737  010000  001372          BIT     #IVC,RMER2I     ;IVC ERROR??
 9991  045710  001427                          BEQ     52$             ;NO!!
 9992  045712  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
 9993  045716  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
 9994  045724  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
 9995  045730  112776  000060  000000          MOVB    #60,@(SP)       ;ERROR 60 IF VV = 0
 9996  045736  032737  000100  001342          BIT     #VV,RMDSI
 9997  045744  001403                          BEQ     51$
 9998  045746  112776  000061  000000          MOVB    #61,@(SP)       ;ERROR 61 IF VV  = 1
 9999  045754  162716  000002          51$:    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10000  045760  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10001  045762  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
10002  045766  000240                          NOP
10003
10004  045770  013737  001372  001142  52$:     MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10005  045776  042737  137777  001142          BIC     #^CSKI,$BDDAT   ;CLEAR DONT CARES
10006  046004  013737  046656  001140          MOV     300$,$GDDAT     ;GET EXPECTED SKI STATUS
10007  046012  042737  137777  001140          BIC     #^CSKI,$GDDAT   ;CLEAR DONT CARES
10008  046020  001417                          BEQ     53$             ;BRANCH IF 0 EXPECTED
10009
10010                                          ;REPORT ERROR IF SKI IS NOT SET (IAE WAS NOT DETECTED)
10011  046022  032737  040000  001142          BIT     #SKI,$BDDAT     ;WAS SKI DETECTED ??
10012  046030  001032                          BNE     54$             ;YES !!
10013  046032  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10014  046036  112776  000267  000000          MOVB    #267,@(SP)      ;WRITE ERROR NUMBER
10015  046044  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10016  046050  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10017  046052  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10018  046056  000443                          BR      6$              ;GO TO NEXT ERROR CHECK
10019  046060                                  53$:
```

```
10020
10021                                   ;REPORT ERROR IF SKI IS SET
10022  046060  032737  040000  001142           BIT     #SKI,$BDDAT     ;IS SKI SET ??
10023  046066  001413                           BEQ     54$             ;NO - SKI IS OK
10024  046070  062716  000004                   ADD     #4,(SP)         ;MOVE (SP) TO ERROR
10025  046074  112776  000054  000000           MOVB    #54,@(SP)       ;LOAD ERROR NUMBER
10026  046102  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10027  046106  004736                           JSR     PC,@(SP)+       ;REPORT SEEK ERROR
10028  046110  162716  000010                   SUB     #10,(SP)        ;RESTORE (SP)
10029  046114  000240                           NOP
10030
10031                                   ;REPORT ANY DEVICE CHECK
10032  046116  032737  000200  001372  54$:     BIT     #DVC,RMER2I     ;WAS THERE DVC DURING SEEK??
10033  046124  001420                           BEQ     6$              ;NO!!
10034  046126  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS
10035  046132  013737  001372  001142           MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10036  046140  062716  000004                   ADD     #4,(SP)
10037  046144  112776  000055  000000           MOVB    #55,@(SP)       ;ERROR  #55
10038  046152  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10039  046156  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10040  046160  162716  000010                   SUB     #10,(SP)        ;RESTORE SP
10041  046164  000240                           NOP
10042
10043                                   ;REPORT ANY "OPI" ERROR AS OPI WITH MOL  =0, OR  OPI
10044                                   ;BECAUSE ON CYLINDER LATCH DIDN'T RESET
10045  046166  032737  020000  001344  6$:      BIT     #OPI,RMER1I     ;"OPI" ERROR??
10046  046174  001427                           BEQ     8$              ;NO!!
10047  046176  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS
10048  046202  013737  001344  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10049  046210  062716  000004                   ADD     #4,(SP)         ;MOVE (SP) TO ERROR
10050  046214  112776  000052  000000           MOVB    #52,@(SP)       ;LOAD ERROR NUMBER
10051  046222  032737  010000  001342           BIT     #MOL,RMDSI      ;WAS MEDIUM ON LINE??
10052  046230  001403                           BEQ     7$              ;NO!!
10053  046232  112776  000053  000000           MOVB    #53,@(SP)       ;YES - CHANGE ERROR NUMBER
10054  046240  162716  000002           7$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10055  046244  004736                           JSR     PC,@(SP)+       ;REPORT "OPI" ERROR
10056  046246  162716  000010                   SUB     #10,(SP)        ;RESTORE (SP)
10057  046252  000240                           NOP
10058
10059                                   ;SEE IF "PIP" IS 0, AND "ATA","MOL"AND"VV" =1
10060  046254  013746  001342           8$:     MOV     RMDSI,-(SP)
10061  046260  042716  047677                   BIC     #^C<ATA!PIP!MOL!VV>,(SP)
10062  046264  022726  110100                   CMP     #ATA!MOL!VV,(SP)+
10063  046270  001002                           BNE     9$              ;ERROR IN RMDS
10064  046272  000137  046626                   JMP     14$             ;RMDS IS OK
10065
10066                                   ;REPORT ERROR IF MOL = 0-AND OPI = 0
10067  046276  032737  010000  001342  9$:      BIT     #MOL,RMDSI      ;IS MOL RESET??
10068  046304  001030                           BNE     10$             ;NO - MOL IS SET
10069  046306  032737  020000  001344           BIT     #OPI,RMER1I     ;WAS OPI SET
10070  046314  001024                           BNE     10$             ;YES - DONT REPORT ERROR
10071  046316  013737  001342  001140           MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10072  046324  052737  010000  001140           BIS     #MOL,$GDDAT
10073  046332  013737  001342  001142           MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10074  046340  062716  000004                   ADD     #4,(SP)
10075  046344  112776  000062  000000           MOVB    #62,@(SP)
```

```
10076  046352  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10077  046356  004736                    JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
10078  046360  162716  000010            SUB     #10,(SP)
10079  046364  000240                    NOP
10080
10081                          ;REPORT AN ERROR IF "PIP" IS STIL SET AND SKI NOT SET
10082  046366  032737  020000  001342  10$:  BIT     #PIP,RMDSI       ;IS "PIP" STILL SET??
10083  046374  001430                    BEQ     11$              ;NO!!
10084  046376  032737  040000  001372      BIT     #SKI,RMER2I      ;WAS "SKI"SET??
10085  046404  001024                    BNE     11$              ;YES-DONT REPORT PIP
10086  046406  013737  001342  001140      MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
10087  046414  042737  020000  001142      BIC     #PIP,$BDDAT
10088  046422  013737  001342  001142      MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
10089  046430  062716  000004              ADD     #4,(SP)          ;MOVE (SP) TO ERROR
10090  046434  112776  000056  000000      MOVB    #56,@(SP)        ;LOAD ERROR NUMBER
10091  046442  162716  000002              SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10092  046446  004736                    JSR     PC,@(SP)+        ;REPORT "PIP" SET AFTER SEEK
10093  046450  162716  000010            SUB     #10,(SP)                 ;RESTORE (SP)
10094  046454  000240                    NOP
10095
10096                          ;REPORT AN ERROR IF "ATA" IS NOT SET
10097  046456  032737  100000  001342  11$:  BIT     #ATA,RMDSI       ;WAS "ATA" SET ??
10098  046464  001024                    BNE     13$              ;YES!!
10099  046466  013737  001342  001140      MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
10100  046474  052737  110600  001140      BIS     #ATA!MOL!DPR!DRY,$GDDAT
10101  046502  013737  001342  001142      MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
10102  046510  062716  000004              ADD     #4,(SP)          ;MOVE (SP) TO ERROR
10103  046514  112776  000057  000000      MOVB    #57,@(SP)        ;LOAD ERROR NUMBER
10104  046522  162716  000002              SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10105  046526  004736                    JSR     PC,@(SP)+        ;REPORT ATTENTION NOT SET DURING
10106                                                           ;SEEK TEST
10107  046530  162716  000010            SUB     #10,(SP)                 ;RESTORE (SP)
10108  046534  000240                    NOP
10109
10110                          ;REPORT ERROR IF VOLUME VALID IS RESET AND IVC IS ZERO
10111  046536  032737  000100  001342  13$:  BIT     #VV,RMDSI        ;IS VV = 0 ??
10112  046544  001030                    BNE     14$              ;NO!!
10113  046546  032737  010000  001372      BIT     #IVC,RMER2I      ;IS IVC ALSO 0 ??
10114  046554  001024                    BNE     14$              ;NO - IVC IS SET
10115  046556  013737  001342  001140      MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
10116  046564  052737  000100  001140      BIS     #VV,$GDDAT
10117  046572  013737  001342  001142      MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
10118  046600  062716  000004              ADD     #4,(SP)
10119  046604  112776  000064  000000      MOVB    #64,@(SP)        ;ERROR #64
10120  046612  162716  000002              SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10121  046616  004736                    JSR     PC,@(SP)+
10122  046620  162716  000010            SUB     #10,(SP)
10123  046624  000240                    NOP
10124  046626                         14$:
10125
10126                          ;MODIFY THE RETURN ADDRESS IF AN ERROR WAS DETECTED
10127  046626  000240                    NOP
10128  046630  062716  000004            ADD     #4,(SP)          ;MOVE (SP) TO ERROR CALL
10129  046634  105776  000000            TSTB    @(SP)            ;WAS ERROR CALLED??
10130  046640  001403                    BEQ     15$              ;NO!!
10131  046642  062716  000004            ADD     #4,(SP)          ;MOVE TO ERROR RETURN
```

```
10132  046646  000402                    BR      16$
10133
10134  046650  162716  000004    15$:    SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10135  046654  000207              16$:    RTS     PC              ;RETURN
10136
10137  046656  000000              300$:   .WORD   0               ;ERROR FLAGS
10138
```

```
10139                                   .SBTTL  CONTROLLER CLEAR SUBROUTINE
10140
10141                                   ;THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTERS,
10142                                   ;AND DRIVES, THEN SELECTS THE DRIVE.
10143                                   ;
10144                                   ;CALL:  JSR     PC,CNTCLR
10145                                   ;       BR      ???             RETURN HERE IF NO ERROR FOUND
10146                                   ;       NOP                     RETURN HERE IF ANY ERROR FOUND
10147                                   ;       ERROR                   SUB DEFINES ERROR NUMBER
10148                                   ;       ???
10149
10150   046660                          CNTCLR:
10151   046660  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10152   046662  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10153   046664  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10154   046670  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10155   046674  012737  046734  000004        MOV     #10$,ERRVEC     ;SETUP FOR BUS TIMEOUT
10156   046702  012737  000300  000006        MOV     #PR6,ERRVEC+2
10157   046710  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
10158   046714  012760  000040  000010        MOV     #CLR,RMCS2(R0)  ;CLEAR MASSBUS
10159   046722  013701  001450                MOV     TSTQUE,R1
10160   046726  111160  000010                MOVB    (R1),RMCS2(R0)  ;SELECT DEVICE
10161   046732  000412                        BR      20$
10162   046734  022626                10$:    CMP     (SP)+,(SP)+     ;ADJUST STACK
10163   046736  062766  000004  000010        ADD     #4,10(SP)       ;MOVE SP TO USER'S ERROR CALL
10164   046744  112776  000007  000010        MOVB    #7,@10(SP)      ;WRITE THE ERROR NUMBER
10165   046752  162766  000002  000010        SUB     #2,10(SP)
10166   046760                          20$:
10167   046760  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10168   046764  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10169   046770  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
10170   046772  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
10171   046774  000207                        RTS     PC
10172
```

```
10173                                   .SBTTL  CONTROLLER CLEAR STATUS CHECK SUBROUTINE
10174
10175                                   ;THIS SUBROUTINE VERIFIES THAT THE RM03 SUBSYSTEM IS INITIALIZED BASED ON
10176                                   ;STATUS STORED IN THE GET BUFFER.  THIS SUBROUTINE SHOULD ONLY BE
10177                                   ;USED FOLLOWING A CONTROLLER CLEAR OPERATION, I.E., WRITING A 1 IN BIT
10178                                   ;5 OF RMCS2, BECAUSE THE ERROR MESSAGES ARE BASED ON THAT CONDITION.
10179
10180                                   ;STATUS PERTINENT TO THE DEVICE IS NOT CHECKED.  IN PARTICULAR, THE
10181                                   ;FOLLOWING STATUS BITS ARE NOT CHECKED:
10182                                   ;
10183                                   ;       ATA,ERR,PIP,MOL,WRL,LBT,PGM,VV,OM,UNS,SKI,DVC
10184                                   ;
10185
10186                                   ;CALL:
10187                                   ;(1)    JSR     PC,CLRSTS
10188                                   ;       BR      ???             RETURN HERE IF NO ERROR
10189                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10190                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10191                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10192                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10193
10194  046776                           CLRSTS:
10195
10196                                   ;CLEAR USER'S ERROR CALL
10197  046776  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO ERROR
10198  047002  105076  000000                   CLRB    @(SP)           ;CLEAR ERROR NUMBER
10199  047006  162716  000004                   SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR
10200                                   ;REPORT ERROR IF RMCS1 NOT INITIALIZED
10201  047012  013737  001330  001142  4$:      MOV     RMCS11,$BDDAT   ;VERIFY RMCS1
10202  047020  042737  100000  001142           BIC     #SC,$BDDAT      ;IGNORE SPECIAL CONDITION
10203  047026  012737  004200  001140           MOV     #DVA!RDY,$GDDAT ;EXPECT DVA & RDY
10204  047034  023737  001140  001142           CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED, RECEIVED
10205  047042  001413                           BEQ     5$              ;BRANCH IF EQUAL
10206  047044  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10207  047050  112776  000126  000000           MOVB    #126,@(SP)      ;WITE ERROR NUMBER IN CALL
10208  047056  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10209  047062  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10210  047064  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10211  047070  000240                           NOP
10212                                   ;REPORT ERROR IF RMBA NOT RESET
10213  047072  005037  001140           5$:     CLR     $GDDAT          ;VERIFY RMBA IS ZERO
10214  047076  013737  001334  001142           MOV     RMBA1,$BDDAT
10215  047104  001413                           BEQ     7$              ;BRANCH IF ZERO
10216  047106  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10217  047112  112776  000127  000000           MOVB    #127,@(SP)      ;WITE ERROR NUMBER IN CALL
10218  047120  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10219  047124  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10220  047126  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10221  047132  000240                           NOP
10222                                   ;REPORT ERROR IF RMCS2 NOT INITIALIZED
10223  047134  013737  001340  001142  7$:      MOV     RMCS21,$BDDAT   ;VERIFY RMCS2
10224  047142  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10225  047144  005046                           CLR     -(SP)           ;EXPECT IR & UNIT NUMBER
10226  047146  013701  001450                   MOV     TSTQUE,R1       ;R1 = ADDRESS OF TEST QUE
10227  047152  111116                           MOVB    (R1),(SP)
10228  047154  052716  000100                   BIS     #IR,(SP)
```

```
10229  047160  012637  001140                  MOV     (SP)+,$GDDAT
10230  047164  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10231  047166  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10232  047174  001413                          BEQ     9$              ;BRANCH IF EQUAL
10233  047176  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10234  047202  112776  000130  000000          MOVB    #130,@(SP)      ;WITE ERROR NUMBER IN CALL
10235  047210  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10236  047214  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10237  047216  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10238  047222  000240                          NOP
10239                          ;REPORT ERROR IF RMER1 NOT RESET-IGNORE UNS
10240  047224  005037  001140          9$:     CLR     $GDDAT          ;VERIFY RMER1
10241  047230  013737  001344  001142          MOV     RMER11,$BDDAT
10242  047236  042737  040000  001142          BIC     #UNS,$BDDAT     ;IGNORE UNSAFE
10243  047244  001413                          BEQ     13$             ;BRANCH IF ZERO
10244  047246  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10245  047252  112776  000131  000000          MOVB    #131,@(SP)      ;WITE ERROR NUMBER IN CALL
10246  047260  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10247  047264  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10248  047266  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10249  047272  000240                          NOP
10250                          ;REPORT ERROR IF RMMR1 NOT INITIALIZED-IGNORE WC,LS,LST
10251  047274  013737  001354  001142  13$:     MOV     RMMR11,$BDDAT   ;VERIFY RMMR
10252  047302  042737  000046  001142          BIC     #WC!LS!LST,$BDDAT       ;IGNORE WORD CLOCK, SCT, TRK
10253  047310  012737  000010  001140          MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA BIT
10254  047316  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
10255  047324  001413                          BEQ     17$             ;BRANCH IF 0
10256  047326  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10257  047332  112776  000133  000000          MOVB    #133,@(SP)      ;WITE ERROR NUMBER IN CALL
10258  047340  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10259  047344  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10260  047346  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10261  047352  000240                          NOP
10262                          ;REPORT AN ERROR IF RMEC2 IS NOT RESET
10263  047354  005037  001140          17$:    CLR     $GDDAT          ;EXPECT ZEROS
10264  047360  013737  001376  001142          MOV     RMEC21,$BDDAT   ;VERIFY RMEC2=0
10265  047366  001413                          BEQ     19$
10266  047370  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10267  047374  112776  000135  000000          MOVB    #135,@(SP)      ;WITE ERROR NUMBER IN CALL
10268  047402  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10269  047406  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10270  047410  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10271  047414  000240                          NOP
10272                          ;REPORT ERROR IF RMMR2 NOT INITIALIZED-IGNORE RQA,RQB
10273  047416  013737  001370  001142  19$:     MOV     RMMR21,$BDDAT   ;VERIFY RMMR2
10274  047424  042737  140000  001142          BIC     #RQA!RQB,$BDDAT
10275  047432  012737  011777  001140          MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
10276  047440  023737  001140  001142          CMP     $GDDAT,$BDDAT
10277  047446  001413                          BEQ     21$
10278  047450  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10279  047454  112776  000136  000000          MOVB    #136,@(SP)      ;WITE ERROR NUMBER IN CALL
10280  047462  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10281  047466  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10282  047470  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10283  047474  000240                          NOP
10284                          ;REPORT ERROR IF RMER2 NOT RESET-IGNORE SKI,DVC
```

```
10285  047476  005037  001140          21$:    CLR     $GDDAT              ;EXPECT ALL ZEROS
10286  047502  013737  001372  001142          MOV     RMER21,$BDDAT       ;VERIFY RMER2
10287  047510  042737  040200  001142          BIC     #SKI!DVC,$BDDAT     ;IGNORE DEVICE ERRORS
10288  047516  001413                          BEQ     215$                ;BRANCH IF OTHER BITS 0
10289  047520  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
10290  047524  112776  000174  000000          MOVB    #174,@(SP)          ;WITE ERROR NUMBER IN CALL
10291  047532  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
10292  047536  004736                          JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
10293  047540  162716  000010                  SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
10294  047544  000240                          NOP
10295                          ;REPORT ERROR IF RMDS NOT INITIALIZED
10296  047546  013737  001342  001142  215$:   MOV     RMDSI,$BDDAT        ;TEST DRIVE STATUS REGISTER
10297  047554  042737  177717  001142          BIC     #^C<DRY!DPR>,$BDDAT
10298  047562  012737  000600  001140          MOV     #DPR!DRY,$GDDAT     ;EXPECTED DRIVE STATUS
10299  047570  023737  001140  001142          CMP     $GDDAT,$BDDAT       ;COMPARE EXPECTED & RECEIVED
10300  047576  001413                          BEQ     22$                 ;BRANCH IF EQUAL
10301  047600  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
10302  047604  112776  000134  000000          MOVB    #134,@(SP)          ;WRITE ERROR NUMBER
10303  047612  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
10304  047616  004736                          JSR     PC,@(SP)+           ;REPORT ERROR TO USER
10305  047620  162716  000010                  SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
10306  047624  000240                          NOP
10307  047626  062716  000004          22$:    ADD     #4,(SP)             ;MOVE SP TO ERROR CALL
10308  047632  105776  000000                  TSTB    @(SP)               ;WAS AN ERROE DETECTED??
10309  047636  001403                          BEQ     23$                 ;NO!!
10310  047640  062716  000004                  ADD     #4,(SP)             ;YES - MOVE TO ERROR RETURN
10311  047644  000402                          BR      24$
10312  047646  162716  000004          23$:    SUB     #4,(SP)             ;MOVE SP TO NO ERROR RETURN
10313  047652  000240                  24$:    NOP
10314  047654  000207                          RTS     PC
```

```
10315                                   .SBTTL  PACK ACKNOWLEDGE STATUS CHECK
10316
10317                                   ;THIS SUBROUTINE CHECKS THE RESULTS OF A PACK ACKNOWLEDGE
10318                                   ;COMMAND USING THE STATUS STORED IN THE GET BUFFER.  ERRORS ARE
10319                                   ;REPORTED TO THE USER VIA THE USER'S ERROR CALL.
10320
10321                                   ;CALL:
10322                                   ;(1)    JSR     PC,ACKSTS
10323                                   ;       BR      ???             RETURN HERE IF NO ERROR
10324                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10325                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10326                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10327                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10328
10329  047656                           ACKSTS:
10330
10331                                   ;CLEAR USER'S ERROR CALL
10332  047656  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10333  047662  105076  000000                  CLRB    @(SP)           ;CLEAR LOW ORDER BYTE
10334  047666  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK
10335
10336                                   ;REPORT AN ERROR  IF "VV" IS 0
10337  047672  032737  000100  001342          BIT     #VV,RMDSI       ;IS VOLUME VALID SET??
10338  047700  001024                          BNE     1$              ;YES!!
10339  047702  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10340  047710  052737  000100  001140          BIS     #VV,$GDDAT
10341  047716  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10342  047724  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10343  047730  112776  000155  000000          MOVB    #155,@(SP)      ;WRITE NUMBER IN ERROR CALL
10344  047736  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10345  047742  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR
10346  047744  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10347  047750  000240                          NOP
10348  047752                           1$:
10349
10350                                   ;REPORT AN ERROR IF "MOL" IS 0
10351  047752  032737  010000  001342          BIT     #MOL,RMDSI      ;IS MOL SET??
10352  047760  001024                          BNE     2$              ;YES!!
10353  047762  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10354  047770  052737  010000  001140          BIS     #MOL,$GDDAT
10355  047776  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10356  050004  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10357  050010  112776  000041  000000          MOVB    #41,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10358  050016  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10359  050022  004736                          JSR     PC,@(SP)+       ;REPORT TH ERROR
10360  050024  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO BRANCH
10361  050030  000240                          NOP
10362  050032                           2$:
10363
10364                                   ;SEE IF "UNS","OPI","RMR","ILR", OR "ILF" IS SET
10365  050032  032737  060007  001344          BIT     #UNS!OPI!RMR!ILR!ILF,RMER1I
10366  050040  001570                          BEQ     7$
10367
10368                                   ;REPORT AN ERROR IF "UNS" IS SET
10369  050042  032737  040000  001344          BIT     #UNS,RMER1I     ;WAS UNS SET??
10370  050050  001424                          BEQ     3$              ;NO!!
```

```
10371  050052  013737  001344  001142          MOV     RMER11,$BDDAT   ;RECEIVED STATUS
10372  050060  013737  001344  001140          MOV     RMER11,$GDDAT   ;EXPECTED STATUS
10373  050066  042737  040000  001140          BIC     #UNS,$GDDAT
10374  050074  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10375  050100  112776  000042  000000          MOVB    #42,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10376  050106  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10377  050112  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10378  050114  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10379  050120  000240                          NOP
10380  050122                          3$:
10381
10382                                  ;REPORT ANY OPI ERROR
10383  050122  032737  020000  001344          BIT     #OPI,RMER11     ;WAS OPI SET ??
10384  050130  001424                          BEQ     4$              ;NO!!
10385  050132  013737  001344  001142          MOV     RMER11,$BDDAT   ;RECEIVED STATUS
10386  050140  013737  001344  001140          MOV     RMER11,$GDDAT   ;EXPECTED STATUS
10387  050146  042737  020000  001140          BIC     #OPI,$GDDAT
10388  050154  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10389  050160  112776  000043  000000          MOVB    #43,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10390  050166  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10391  050172  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10392  050174  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10393  050200  000240                          NOP
10394  050202                          4$:
10395
10396                                  ;REPORT ANY RMR ERROR
10397  050202  032737  000004  001344          BIT     #RMR,RMER11     ;WAS RMR SET??
10398  050210  001424                          BEQ     5$              ;NO!!
10399  050212  013737  001344  001142          MOV     RMER11,$BDDAT   ;RECEIVED STATUS
10400  050220  013737  001344  001140          MOV     RMER11,$GDDAT   ;EXPECTED STATUS
10401  050226  042737  000004  001140          BIC     #RMR,$GDDAT
10402  050234  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10403  050240  112776  000044  000000          MOVB    #44,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10404  050246  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10405  050252  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10406  050254  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10407  050260  000240                          NOP
10408  050262                          5$:
10409
10410                                  ;REPORT ANY     ILR ERROR
10411  050262  032737  000002  001344          BIT     #ILR,RMER11     ;WAS ILR SET??
10412  050270  001424                          BEQ     6$              ;NO!!
10413  050272  013737  001344  001142          MOV     RMER11,$BDDAT   ;RECEIVED STATUS
10414  050300  013737  001344  001140          MOV     RMER11,$GDDAT   ;EXPECTED STATUS
10415  050306  042737  000002  001140          BIC     #ILR,$GDDAT
10416  050314  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10417  050320  112776  000045  000000          MOVB    #45,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10418  050326  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10419  050332  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10420  050334  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10421  050340  000240                          NOP
10422  050342                          6$:
10423
10424                                  ;REPORT ANY     ILF ERROR
10425  050342  032737  000001  001344          BIT     #ILF,RMER11     ;WAS ILF SET??
10426  050350  001424                          BEQ     7$              ;NO!!
```

```
10427  050352  013737  001344  001142          MOV     RMER11,$BDDAT   ;RECEIVED STATUS
10428  050360  013737  001344  001140          MOV     RMER11,$GDDAT   ;EXPECTED STATUS
10429  050366  042737  000001  001140          BIC     #1LF,$GDDAT
10430  050374  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10431  050400  112776  000046  000000          MOVB    #46,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10432  050406  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10433  050412  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10434  050414  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10435  050420  000240                          NOP
10436  050422                          7$:
10437
10438                                  ;AUGMENT RETURN ADDRESS IF ERROR WAS FOUND
10439  050422  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10440  050426  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
10441  050432  001403                          BEQ     8$              ;NO!!
10442  050434  062716  000004                  ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
10443  050440  000402                          BR      9$
10444  050442  162716  000004          8$:     SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
10445  050446  000240                  9$:     NOP
10446  050450  000207                          RTS     PC
10447
```

```
10448                                   .SBTTL  RECALIBRATE STATUS CHECK SUBROUTINE
10449
10450                                   ;THIS SUBROUTINE CHECKS THE RESULTS OF A RECALIBRATE OPERATION
10451                                   ;USING THE STATUS STORED IN THE GET BUFFER.
10452
10453                                   ;CALL:
10454                                   ;
10455                                   ;(1)    JSR     PC,RCLSTS       ;CALL SUBROUTINE
10456                                   ;       BR      ???             RETURN HERE IF NO ERROR
10457                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10458                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10459                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10460                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10461
10462  050452                           RCLSTS:
10463
10464                                   ;CLEAR USER'S ERROR NUMBER
10465  050452  062716  000004                  ADD     #4,(SP)
10466  050456  105076  000000                  CLRB    @(SP)           ;CLEAR USER'S ERROR CALL
10467  050462  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO BRANCH
10468
10469
10470                                   ;SEE IF "PAR" OR "ILF" OR "OPI" OR "IAE" IS SET
10471  050466  032737  022011  001344          BIT     #OPI!PAR!ILF!IAE,RMER1I
10472  050474  001553                          BEQ     4$              ;NONE ARE SET - GO TO NEXT CHECK
10473
10474                                   ;REPORT ANY MASSBUS CONTROL BUS PARITY ERROR, I.E.,
10475                                   ;"PAR" = 1 AND "DPE" = 0
10476  050476  032737  000010  001344          BIT     #PAR,RMER1I     ;WAS "PAR" SET??
10477  050504  001430                          BEQ     1$              ;NO!!
10478  050506  032737  000010  001372          BIT     #DPE,RMER2I     ;WAS "DPE" SET??
10479  050514  001024                          BNE     1$              ;YES - NOT A REGISTER ERROR
10480  050516  013737  001344  001140          MOV RMER1I,$GDDAT       ;EXPECTED STATUS
10481  050524  042737  000010  001140          BIC     #PAR,$GDDAT
10482  050532  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10483  050540  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10484  050544  112776  000050  000000          MOVB    #50,@(SP)       ;WRITE ERROR NUMBER IN CALL
10485  050552  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10486  050556  004736                          JSR     PC,@(SP)+       ;GO REPORT ERROR
10487  050560  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10488  050564  000240                          NOP
10489  050566                           1$:
10490
10491                                   ;REPORT ANY "ILF" ERROR
10492  050566  032737  000001  001344          BIT     #ILF,RMER1I     ;WAS "ILF" SET??
10493  050574  001424                          BEQ     2$              ;NO!!
10494  050576  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10495  050604  042737  000001  001140          BIC     #ILF,$GDDAT
10496  050612  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10497  050620  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10498  050624  112776  000071  000000          MOVB    #71,@(SP)       ;WRITE ERROR NUMBER IN CALL
10499  050632  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10500  050636  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10501  050640  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10502  050644  000240                          NOP
10503  050646                           2$:
```

```
10504
10505                                   ;REPORT ANY "OPI" ERROR AS
10506                                   ;       . OPI DUE TO "MOL" = 0
10507                                   ;       . OPI BECAUSE ON CYLINDER LATCH DIDN'T RESET
10508  050646  032737  020000  001344           BIT     #OPI,RMER1I     ;WAS OPI SET??
10509  050654  001433                           BEQ     31$             ;NO!!
10510  050656  013737  001344  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10511  050664  042737  020000  001140           BIC     #OPI,$GDDAT
10512  050672  013737  001344  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10513  050700  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10514  050704  112776  000072  000000           MOVB    #72,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
10515  050712  032737  010000  001342           BIT     #MOL,RMDSI      ;WAS "MOL" = 0??
10516  050720  001403                           BEQ     3$              ;YES!!
10517  050722  112776  000073  000000           MOVB    #73,@(SP)       ;NO - CHANGE ERROR NUMBER
10518  050730  162716  000002           3$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10519  050734  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10520  050736  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10521  050742  000240                           NOP
10522  050744                           31$:
10523
10524                                   ;REPORT AN ERROR IF "IAE" IS SET
10525  050744  032737  002000  001344           BIT     #IAE,RMER1I     ;IS "IAE" SET??
10526  050752  001424                           BEQ     4$              ;NO!!
10527  050754  013737  001344  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10528  050762  042737  002000  001140           BIC     #IAE,$GDDAT
10529  050770  013737  001344  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10530  050776  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10531  051002  112776  000070  000000           MOVB    #70,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
10532  051010  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10533  051014  004736                           JSR     PC,@(SP)+       ;REPORT ERROR
10534  051016  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
10535  051022  000240                           NOP
10536  051024                           4$:
10537
10538                                   ;SEE IF "SKI" OR "IVC" OR "DVC" IS SET
10539  051024  032737  050200  001372           BIT     #SKI!IVC!DVC,RMER2I
10540  051032  001517                           BEQ     8$              ;NONE OF THE BITS ARE SET
10541
10542
10543                                   ;REPORT ANY "IVC" ERROR AS
10544                                   ;       .IVC WITH VV = 0
10545                                   ;       .ERRONEOUS IVC ERROR
10546  051034  032737  010000  001372           BIT     #IVC,RMER2I     ;WAS IVC SET??
10547  051042  001433                           BEQ     6$              ;NO!!
10548  051044  013737  001372  001140           MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
10549  051052  042737  010000  001140           BIC     #IVC,$GDDAT
10550  051060  013737  001372  001142           MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10551  051066  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10552  051072  112776  000074  000000           MOVB    #74,@(SP)       ;;WRITE ERROR NUMBER IN CALL
10553  051100  032737  000100  001342           BIT     #VV,RMDSI       ;WAS VV = 0??
10554  051106  001403                           BEQ     5$              ;YES!!
10555  051110  112776  000075  000000           MOVB    #75,@(SP)       ;NO - CHANGE ERROR NUMBER
10556  051116  162716  000002           5$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10557  051122  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10558  051124  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10559  051130  000240                           NOP
```

```
10560  051132                                  6$:
10561
10562                                          ;REPORT ANY "SKI" ERROR
10563  051132  032737  040000  001372                  BIT     #SKI,RMER21     ;WAS SKI SET??
10564  051140  001424                                  BEQ     7$              ;NO!!
10565  051142  013737  001372  001140                  MOV     RMER21,$GDDAT   ;EXPECTED STATUS
10566  051150  042737  040000  001140                  BIC     #SKI,$GDDAT
10567  051156  013737  001372  001142                  MOV     RMER21,$BDDAT   ;RECEIVED STATUS
10568  051164  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10569  051170  112776  000076  000000                MOVB    #76,@(SP)       ;WRITE ERROR NUMBER
10570  051176  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10571  051202  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10572  051204  162716  000010                        SUB     #10,(SP)        ;MOVE SP TO BRANCH
10573  051210  000240                                  NOP
10574  051212                                  7$:
10575
10576                                          ;REPORT ANY "DVC" ERROR
10577  051212  032737  000200  001372                  BIT     #DVC,RMER21     ;WAS "DVC" SET??
10578  051220  001424                                  BEQ     8$              ;NO!!
10579  051222  013737  001372  001140                  MOV     RMER21,$GDDAT   ;EXPECTED STATUS
10580  051230  042737  000200  001140                  BIC     #DVC,$GDDAT
10581  051236  013737  001372  001142                  MOV     RMER21,$BDDAT   ;RECEIVED STATUS
10582  051244  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10583  051250  112776  000077  000000                MOVB    #77,@(SP)       ;WRITE ERROR NUMBER
10584  051256  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10585  051262  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10586  051264  162716  000010                        SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
10587  051270  000240                                  NOP
10588  051272                                  8$:
10589
10590                                          ;SEE IF "PIP" AND "OM" ARE 0, AND "ATA","MOL" AND "VV" ARE 1
10591  051272  013746  001342                          MOV     RMDSI,-(SP)     ;PUT RMDS ON STACK
10592  051276  042716  047676                          BIC     #^C<PIP!MOL!VV!OM!ATA>,(SP)
10593  051302  022726  110100                          CMP     #ATA!MOL!VV,(SP)+
10594  051306  001002                                  BNE     85$
10595  051310  000137  051724                          JMP     13$
10596  051314                                  85$:
10597
10598                                          ;REPORT AN ERROR IF MOL = 0 AND OPI = 0, I.E., MEDIUM WENT OFF
10599                                          ;LINE AFTER RECALIBRATE WAS INITIATED
10600  051314  032737  010000  001342                  BIT     #MOL,RMDSI      ;DID MOL DROP??
10601  051322  001030                                  BNE     9$              ;NO!!
10602  051324  032737  020000  001344                  BIT     #OPI,RMER11     ;WAS OPI ERROR REPORTED??
10603  051332  001024                                  BNE     9$              ;YES - DON'T REPORT MOL=0
10604  051334  013737  001342  001140                  MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10605  051342  052737  010000  001140                  BIS     #MOL,$GDDAT
10606  051350  013737  001342  001142                  MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10607  051356  062716  000004                        ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10608  051362  112776  000100  000000                MOVB    #100,@(SP)      ;WRITE ERROR NUMBER
10609  051370  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10610  051374  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10611  051376  162716  000010                        SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
10612  051402  000240                                  NOP
10613  051404                                  9$:
10614
10615                                          ;REPORT AN ERROR IF "VV" = 0 AND "IVC" = 0
```

```
10616  051404  032737  000100  001342          BIT     #VV,RMDSI       ;DID "VV" DROP??
10617  051412  001030                          BNE     10$             ;NO!!
10618  051414  032737  010000  001372          BIT     #IVC,RMER2I     ;WAS THERE A IVC ERROR??
10619  051422  001024                          BNE     10$             ;YES - DONT REPORT VV=0
10620  051424  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10621  051432  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10622  051440  052737  000100  001140          BIS     #VV,$GDDAT
10623  051446  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10624  051452  112776  000101  000000          MOVB    #101,@(SP)      ;WRITE ERROR NUMBER IN CALL
10625  051460  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10626  051464  004736                          JSR     PC,@(SP)+
10627  051466  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
10628  051472  000240                          NOP
10629  051474                          10$:
10630
10631                                  ;REPORT AN ERROR IF ATA IS NOT SET
10632  051474  032737  100000  001342          BIT     #ATA,RMDSI      ;WAS ATA SET DURING RECALIBRATE??
10633  051502  001024                          BNE     11$             ;YES!!
10634  051504  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10635  051512  052737  100000  001140          BIS     #ATA,$GDDAT
10636  051520  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10637  051526  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10638  051532  112776  000102  000000          MOVB    #102,@(SP)      ;WRITE ERROR NUMBER IN CALL
10639  051540  162716  000002                  SUB     #2,(SP)
10640  051544  004736                          JSR     PC,@(SP)+
10641  051546  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
10642  051552  000240                          NOP
10643
10644  051554                          11$:
10645
10646                                  ;REPORT AN ERROR IF "OM" IS NOT ZERO BECAUSE RECALIBRATE SHOULD
10647                                  ;ALWAYS CLEAR OFFSET MODE
10648  051554  032737  000001  001342          BIT     #OM,RMDSI       ;WAS "OM" RESET??
10649  051562  001424                          BEQ     12$             ;YES!!
10650  051564  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10651  051572  042737  000001  001140          BIC     #OM,$GDDAT
10652  051600  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10653  051606  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10654  051612  112776  000103  000000          MOVB    #103,@(SP)      ;WRITE ERROR NUMBER
10655  051620  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10656  051624  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10657  051626  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
10658  051632  000240                          NOP
10659  051634                          12$:
10660
10661                                  ;REPORT AN ERROR IF "PIP" IS STIL ON, I.E., DRIVE NOT ON
10662                                  ;CYLINDER
10663  051634  032737  020000  001342          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
10664  051642  001430                          BEQ     13$             ;NO!!
10665  051644  032737  040000  001372          BIT     #SKI,RMER2I     ;WAS "SKI" DETECTED??
10666  051652  001024                          BNE     13$             ;YES-DONT REPORT "PIP"
10667  051654  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10668  051662  042737  020000  001140          BIC     #PIP,$GDDAT
10669  051670  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10670  051676  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10671  051702  112776  000104  000000          MOVB    #104,@(SP)      ;WRITE ERROR NUMBER
```

```
10672  051710  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10673  051714  004736                          JSR     PC,@(SP)+
10674  051716  162716  000010                  SUB     #10,(SP)         ;MOVE SP BACK TO USER'S BRANCH
10675  051722  000240                          NOP
10676  051724                          13$:
10677
10678                                  ;SEE IF "ILR" OR "RMR" OR "UNS" IS SET
10679  051724  032737  040006  001344          BIT     #ILR!RMR!UNS,RMER1I
10680  051732  001514                          BEQ     16$
10681
10682                                  ;REPORT AN ERROR IF "ILR" IS SET
10683  051734  032737  000002  001344          BIT     #ILR,RMER1I      ;WAS ILR SET DURING RECALIBRATE??
10684  051742  001424                          BEQ     14$              ;NO!!
10685  051744  013737  001344  001140          MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
10686  051752  042737  000002  001140          BIC     #ILR,$GDDAT
10687  051760  013737  001344  001142          MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
10688  051766  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10689  051772  112776  000105  000000          MOVB    #105,@(SP)       ;WRITE ERROR NUMBER IN CALL
10690  052000  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10691  052004  004736                          JSR     PC,@(SP)+
10692  052006  162716  000010                  SUB     #10,(SP)         ;MOVE SP TO USER'S BRANCH
10693  052012  000240                          NOP
10694  052014                          14$:
10695
10696                                  ;REPORT AN ERROR IF "RMR" IS SET
10697  052014  032737  000004  001344          BIT     #RMR,RMER1I      ;WAS RMR SET??
10698  052022  001424                          BEQ     15$              ;NO!!
10699  052024  013737  001344  001140          MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
10700  052032  042737  000004  001140          BIC     #RMR,$GDDAT
10701  052040  013737  001344  001142          MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
10702  052046  062716  000004                  ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
10703  052052  112776  000106  000000          MOVB    #106,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
10704  052060  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10705  052064  004736                          JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
10706  052066  162716  000010                  SUB     #10,(SP)         ;MOVE SP TO USER'S BRANCH
10707  052072  000240                          NOP
10708  052074                          15$:
10709
10710                                  ;REPORT AN ERROR IF "UNS" IS SET AND "DVC" IS 0
10711  052074  032737  040000  001344          BIT     #UNS,RMER1I      ;WAS UNSAFE ON??
10712  052102  001430                          BEQ     16$              ;NO!!
10713  052104  032737  000200  001372          BIT     #DVC,RMER2I      ;WAS THERE A DEVICE CHECK??
10714  052112  001024                          BNE     16$              ;YES - DON'T REPORT UNSAFE
10715  052114  013737  001344  001140          MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
10716  052122  042737  040000  001140          BIC     #UNS,$GDDAT
10717  052130  013737  001344  001142          MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
10718  052136  062716  000004                  ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
10719  052142  112776  000107  000000          MOVB    #107,@(SP)       ;WRITE ERROR NUMBER
10720  052150  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10721  052154  004736                          JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
10722  052156  162716  000010                  SUB     #10,(SP)         ;MOVE SP BACK TO USER'S BRANCH
10723  052162  000240                          NOP
10724  052164                          16$:
10725
10726                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
10727  052164  062716  000004                  ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
```

```
10728  052170  105776  000000          TSTB    @(SP)           ;WAS AN ERROR REPORTED??
10729  052174  001403                  BEQ     17$             ;NO!!
10730  052176  062716  000004          ADD     #4,(SP)         ;YES - AUGMENT SP RETURN
10731  052202  000402                  BR      18$
10732  052204  162716  000004  17$:     SUB     #4,(SP) ;NO ERROR - RETURN SP TO BRANCH
10733  052210  000240          18$:    NOP
10734  052212  000207                  RTS     PC              ;STATUS CECK IS COMPLETE
10735
```

```
10736                                   .SBTTL  DRIVE CLEAR STATUS CHECK SUBROUTINE
10737                                   ;       BR      ???             RETURN HERE IF NO ERROR
10738                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10739                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10740                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10741                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10742
10743   052214                          DRVSTS:
10744
10745                                   ;CLEAR USER'S ERROR CALL
10746   052214  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10747   052220  105076  000000                 CLRB    @(SP)           ;CLEAR ERROR CALL
10748   052224  162716  000004                 SUB     #4,(SP)         ;MOVE SP TO USER'S BRANCH
10749                                   ;REPORT ERROR IF RMCS1 NOT INITIALIZED
10750   052230  013737  001330  001142 4$:     MOV     RMCS1I,$BDDAT   ;CHECK RMCS1
10751   052236  042737  173700  001142         BIC     #^C<DVA!FNCMSK>,$BDDAT  ;CLEAR DONT CARES
10752   052244  012737  004010  001140         MOV     #DVA!DRVCLR,$GDDAT      ;EXPECT DVA
10753   052252  023737  001140  001142         CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10754   052260  001443                         BEQ     6$              ;BRANCH IF EQUAL
10755   052262  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10756   052266  112776  000141  000000         MOVB    #141,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10757   052274  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10758   052300  004736                         JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10759   052302  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10760   052306  000240                         NOP
10761                                   ;REPORT ERROR IF RMDS NOT INITIALIZED
10762   052310  013737  001342  001142 5$:     MOV     RMDSI,$BDDAT    ;CHECK RMDS
10763   052316  042737  021101  001142         BIC     #PGM!OM!VV!PIP,$BDDAT   ;CLEAR DONT CARES
10764   052324  012737  010600  001140         MOV     #MOL!DPR!DRY,$GDDAT     ;EXPECT DRY & DPR
10765   052332  023737  001140  001142         CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10766   052340  001413                         BEQ     6$              ;BRANCH IF EQUAL
10767   052342  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10768   052346  112776  000142  000000         MOVB    #142,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10769   052354  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10770   052360  004736                         JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10771   052362  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10772   052366  000240                         NOP
10773                                   ;REPORT ERROR IF RMER1 NOT INITIALIZED
10774   052370  005037  001140         6$:     CLR     $GDDAT          ;EXPECT 0'S
10775   052374  013737  001344  001142         MOV     RMER1I,$BDDAT   ;CHECK RMER1
10776   052402  001413                         BEQ     8$              ;BRANCH IF EQUAL
10777   052404  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10778   052410  112776  000143  000000         MOVB    #143,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10779   052416  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10780   052422  004736                         JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10781   052424  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10782   052430  000240                         NOP
10783                                   ;REPORT ERROR IF ATA NOT INITIALIZED
10784   052432  013737  001346  001142 8$:     MOV     RMASI,$BDDAT    ;CHECK ATTENTION BIT
10785   052440  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10786   052442  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10787   052444  013701  001450                 MOV     TSTQUE,R1
10788   052450  116102  000001                 MOVB    1(R1),R2
10789   052454  042702  177400                 BIC     #^CATNMSK,R2
10790   052460  005102                         COM     R2
10791   052462  040237  001142                 BIC     R2,$BDDAT
```

```
10792  052466  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
10793  052470  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10794  052472  005737  001142                  TST     $BDDAT          ;IS ATTENTION CLEARED??
10795  052476  001413                          BEQ     9$              ;BRANCH IF ATTENTION CLEARED
10796  052500  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10797  052504  112776  000144  000000          MOVB    #144,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10798  052512  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10799  052516  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10800  052520  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10801  052524  000240                          NOP
10802                          ;REPORT ERROR IF RMMR1 NOT INITIALIZED
10803  052526  013737  001354  001142  9$:     MOV     RMMR1I,$BDDAT   ;CHECK RMMR
10804  052534  042737  000046  001142          BIC     #WC!LS!LST,$BDDAT       ;CLEAR DONT CARES
10805  052542  012737  000010  001140          MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA ON
10806  052550  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
10807  052556  001413                          BEQ     11$             ;BRANCH IF ZERO
10808  052560  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10809  052564  112776  000145  000000          MOVB    #145,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10810  052572  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10811  052576  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10812  052600  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10813  052604  000240                          NOP
10814                          ;REPORT ERROR IF RMMR2 NOT INITIALIZED
10815  052606  013737  001370  001142  11$:     MOV     RMMR2I,$BDDAT   ;CHECK RMMR2
10816  052614  042737  140000  001142          BIC     #RQA!RQB,$BDDAT ;CLEAR REQA, REQB
10817  052622  012737  011777  001140          MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
10818  052630  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10819  052636  001413                          BEQ     15$             ;BRANCH IF EQUAL
10820  052640  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10821  052644  112776  000146  000000          MOVB    #146,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10822  052652  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10823  052656  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10824  052660  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10825  052664  000240                          NOP
10826  052666  005037  001140          15$:    CLR     $GDDAT          ;EXPECT ZEROS
10827                          ;REPORT ERROR IF RMEC2 NOT RESET
10828  052672  013737  001376  001142          MOV     RMEC2I,$BDDAT   ;CHECK RMEC2
10829  052700  001413                          BEQ     17$             ;BRANCH IF 0
10830  052702  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10831  052706  112776  000150  000000          MOVB    #150,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10832  052714  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10833  052720  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10834  052722  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10835  052726  000240                          NOP
10836                          ;REPORT ERROR IF RMER2 NOT RESET
10837  052730  013737  001372  001142  17$:     MOV     RMER2I,$BDDAT   ;CHECK RMER2
10838  052736  001413                          BEQ     18$             ;BRANCH IF NO ERROR
10839  052740  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10840  052744  112776  000147  000000          MOVB    #147,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10841  052752  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10842  052756  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10843  052760  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10844  052764  000240                          NOP
10845  052766                          18$:
10846
10847  052766                          19$:
```

```
10848
10849                                   ;AUGMENT RETURN ADDRESS IF ANY ERROR WAS FOUND
10850  052766  062716  000004                ADD     #4,(SP)          ;MOVE SP TO ERROR CALL
10851  052772  105776  000000                TSTB    @(SP)            ;WAS AN ERROR DETECTED??
10852  052776  001403                         BEQ     21$              ;NO!!
10853  053000  062716  000004                ADD     #4,(SP)          ;YES - MOVE SP TO ERROR RETURN
10854  053004  000402                         BR      23$
10855  053006  162716  000004        21$:    SUB     #4,(SP) ;MOVE SP BACK TO NO ERROR RETURN
10856  053012  000240                 23$:    NOP
10857  053014  000207                         RTS     PC               ;RETURN TO USER
```

```
10858                                   .SBTTL  DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE
10859
10860                                   ;THIS SUBROUTINE VERIFIES THE RESULTS OF ALL DATA TRANSFER COMMANDS
10861                                   ;USING STATUS STORED IN THE GET BUFFER AND TEST PARAMETERS
10862                                   ;STORED IN THE PUT BUFFER.  ERRORS ARE REPORTED BY WRITING
10863                                   ;THE ERROR NUMBER IN THE USERS ERROR CALL.
10864
10865                                   ;USER'S SUBROUTINE CALL:
10866                                   ;(1)    JSR     PC,DTASTS
10867                                   ;(2)    BR      ??              RETURN HERE IF NO DATA ERRORS
10868                                   ;(3)    NOP                     RETURN HERE TO REPORT AN ERROR
10869                                   ;(4)    ERROR                   SUB WRITES ERROR NUMBER
10870                                   ;(5)    JSR     PC,@(SP)+       USER RETURNS FOR MORE CHECKS
10871                                   ;(6)    ??                      SUB RETURNS HERE AFTER ALL
10872                                   ;                               ERRORS ARE REPORTED
10873
10874  053016                           DTASTS:
10875
10876                                   ;CLEAR USER'S ERROR CALL AND ERROR FLAGS
10877  053016  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
10878  053022  105076  000000                  CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF TRAP
10879  053026  162716  000004                  SUB     #4,(SP)         ;RESTORE SP TO NO ERROR
10880  053032  005037  056412                  CLR     500$            ;CLEAR ERROR FLAGS
10881
10882                                   ;REPORT ANY CONTROL BUS PARITY ERROR WHILE READING REMOTE REGISTERS, I.E., MCPE = 1
10883  053036  032737  020000  001330          BIT     #MCPE,RMCS1I    ;WAS THERE A PARITY ERROR??
10884  053044  001422                          BEQ     10$             ;NO!!
10885  053046  013737  001330  001140          MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
10886  053054  042737  020000  001140          BIC     #MCPE,$GDDAT
10887  053062  013737  001330  001142          MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
10888  053070  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10889  053074  112776  000013  000000          MOVB    #13,@(SP)       ;WRITE ERROR NUMBER
10890  053102  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10891  053106  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10892  053110  000466                          BR      30$
10893  053112                           10$:
10894
10895                                   ;REPORT ANY CONTROL BUS PARITY ERROR WHILE WRING REMOTE REGISTERS, I.E.,
10896                                   ;PAR=1  AND DPE = 0
10897  053112  032737  000010  001344          BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
10898  053120  001435                          BEQ     20$             ;NO!!
10899  053122  032737  000010  001372          BIT     #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
10900  053130  001031                          BNE     20$             ;NO!!
10901  053132  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10902  053140  042737  000010  001140          BIC     #PAR,$GDDAT
10903  053146  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10904  053154  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
10905  053160  112776  000050  000000          MOVB    #50,@(SP)       ;WRITE ERROR NUMBER
10906  053166  032737  001000  001330          BIT     #MXF,RMCS1I     ;DID MXF GET SET??
10907  053174  001003                          BNE     15$             ;YES!!
10908  053176  112776  000274  000000          MOVB    #274,@(SP)      ;NO - CHANGE ERROR NUMBER
10909  053204  162716  000002           15$:   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10910  053210  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10911  053212  000425                          BR      30$
10912
10913  053214                           20$:
```

```
10914
10915                                   ;LOOK FOR ANY ERRORS WHICH MAY HAVE OCCURRED DURING COMMAND INITIATION OR
10916                                   ;MECHANICAL POSITIONING
10917
10918                                   ;FIRST TEST MXF WHICH WOULD INDICATE COMPOSITE ERROR SET WHEN FUNCTION
10919                                   ;CODE AND GO BIT WERE LOADED
10920  053214  032737  001000  001340           BIT     #MXF,RMCS2I     ;WAS "MISSED TRANSFER" SET??
10921  053222  001425                           BEQ     40$             ;NO!!
10922  053224  013737  001340  001140           MOV     RMCS2I,$GDDAT   ;EXPECTED STATUS
10923  053232  042737  001000  001140           BIC     #MXF,$GDDAT
10924  053240  013737  001340  001142           MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
10925  053246  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10926  053252  112776  000275  000000           MOVB    #275,@(SP)      ;WRITE ERROR NUMBER
10927  053260  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10928  053264  004736                           JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10929  053266                          30$:
10930
10931                                   ;RESTORE SP TO NO ERROR RETURN AND BYPASS FURHTER STATUS CHECKING
10932  053266  162716  000010                   SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10933  053272  000137  056364                   JMP     380$            ;SKIP TO END OF SUB
10934
10935  053276                          40$:
10936
10937                                   ;REPORT AN ERROR IF "OPI" ERROR OCCURRED DUE TO "MOL" = 0, OR IF "OPI"
10938                                   ;AND "MOL" ARE SET, BUT "VV" IS RESET, INDICATING AN INTERMITTENT
10939                                   ;"MOL"
10940  053276  032737  020000  001344           BIT     #OPI,RMER1I     ;IS "OPI" SET??
10941  053304  001447                           BEQ     60$             ;NO!!
10942  053306  013737  001344  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10943  053314  042737  020000  001140           BIC     #OPI,$GDDAT
10944  053322  013737  001344  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10945  053330  032737  010000  001342           BIT     #MOL,RMDSI      ;WAS MEDIUM OFF LINE??
10946  053336  001404                           BEQ     45$             ;YES!!
10947  053340  032737  000100  001342           BIT     #VV,RMDSI       ;WAS "MOL" INTERMITTENT??
10948  053346  001013                           BNE     50$             ;NO!!
10949  053350  062716  000004          45$:     ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10950  053354  112776  000276  000000           MOVB    #276,@(SP)      ;WRITE ERROR NUMBER IN CALL
10951  053362  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10952  053366  004736                           JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10953  053370  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10954  053374  000413                           BR      60$
10955  053376                          50$:
10956
10957                                   ;REPORT "OPI" ERROR, WHICH IS DUE TO "ON CYLINDER" NOT DROPPING OR
10958                                   ;"RUN" TIMEOUT (20 MS) OR SEARCH TIMEOUT (50 MS)
10959  053376  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10960  053402  112776  000277  000000           MOVB    #277,@(SP)      ;WRITE ERROR NUMBER IN CALL
10961  053410  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10962  053414  004736                           JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10963  053416  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10964  053422  000240                           NOP
10965  053424                          60$:
10966
10967                                   ;LOOK FOR "IVC" ERROR DURING COMMAND INITIATION
10968  053424  032737  010000  001372           BIT     #IVC,RMER2I     ;WAS THERE AN "IVC" ERROR??
10969  053432  001432                           BEQ     70$             ;NO!!
```

```
10970
10971  053434  013737  001372  001140          ;REPORT "IVC" ERROR DUE TO "VV" = 0, OR REPORT ERRONEOUS "IVC" ERROR
10972  053442  042737  010000  001140                  MOV     RMER21,$GDDAT   ;EXPECTED STATUS
10973  053450  013737  001372  001142                  BIC     #IVC,$GDDAT
10974  053456  062716  000004                          MOV     RMER21,$BDDAT   ;RECEIVED STATUS
10975  053462  112776  000300  000000                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
10976  053470  032737  000100  001342                  MOVB    #300,@(SP)      ;WRITE ERROR NUMBER IN CALL
10977  053476  001403                                  BIT     #VV,RMDSI       ;WAS VOLUME VALID??
10978  053500  112776  000301  000000                  BEQ     65$             ;NO!!
10979  053506  162716  000002                  65$:    MOVB    #301,@(SP)      ;CHANGE ERROR NUMBER
10980  053512  004736                                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10981  053514  162716  000010                          JSR     PC,@(SP)+       ;REPORT "IVC" ERROR AND RETURN
10982  053520                                  70$:    SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10983
10984                                          ;SEE IF "ILF" OR "RMR" IS SET
10985  053520  032737  000007  001344                  BIT     #ILR!ILF!RMR, RMER1I
10986  053526  001510                                  BEQ     100$            ;NO ERRORS DETECTED
10987                                          ;REPORT AN ERROR IF "ILR" IS SET
10988  053530  032737  000002  001344                  BIT     #ILR,RMER1I     ;WAS "ILR" DETECTED??
10989  053536  001424                                  BEQ     80$             ;NO!!
10990  053540  013737  001344  001140                  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10991  053546  042737  000002  001140                  BIC     #ILR,$GDDAT
10992  053554  013737  001344  001142                  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10993  053562  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10994  053566  112776  000302  000000                  MOVB    #302,@(SP)      ;WRITE ERROR NUMBER IN CALL
10995  053574  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10996  053600  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10997  053602  162716  000010                          SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10998  053606  000240                                  NOP
10999  053610                                  80$:
11000
11001                                          ;REPORT AN ERROR IF "ILF" IS SET
11002  053610  032737  000001  001344                  BIT     #ILF,RMER1I     ;WAS "ILF" DETECTED??
11003  053616  001424                                  BEQ     90$             ;NO!!
11004  053620  013737  001344  001140                  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11005  053626  042737  000001  001140                  BIC     #ILF,$GDDAT
11006  053634  013737  001344  001142                  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11007  053642  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11008  053646  112776  000303  000000                  MOVB    #303,@(SP)      ;WRITE ERROR NUMBER IN CALL
11009  053654  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11010  053660  004736                                  JSR     PC,@(SP)+       ;REPORT ERORR AND RETURN
11011  053662  162716  000010                          SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11012  053666  000240                                  NOP
11013  053670                                  90$:
11014
11015                                          ;REPORT AN ERROR IF "RMR" IS SET
11016  053670  032737  000004  001344                  BIT     #RMR,RMER1I     ;WAS "RMR" DETECTED??
11017  053676  001424                                  BEQ     100$            ;NO!!
11018  053700  013737  001344  001140                  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11019  053706  042737  000004  001140                  BIC     #RMR,$GDDAT
11020  053714  013737  001344  001142                  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11021  053722  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11022  053726  112776  000304  000000                  MOVB    #304,@(SP)      ;WRITE ERROR NUMBER IN CALL
11023  053734  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11024  053740  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11025  053742  162716  000010                          SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
```

```
11026  053746  000240                          NOP
11027  053750                          100$:
11028                                  ;DETERMINE WHETHER OR NOT "IAE" SHOULD BE SET AND CHECK FOR ERROR
11029  053750  012737  002000  001140          MOV     #IAE,$GDDAT     ;SETUP FOR "IAE" = 1
11030  053756  052737  040000  056412          BIS     #SKI,500$       ;SET SKI FLAG
11031  053764  022737  001466  001434          CMP     #822.,RMDC0     ;IS CYLINDER > 822??
11032  053772  103425                          BLO     110$            ;YES!!
11033  053774  042737  040000  056412          BIC     #SKI,500$       ;RESET SKI FLAG
11034  054002  122737  000004  001407          CMPB    #4,RMDA0+1      ;IS TRACK > 4??
11035  054010  103416                          BLO     110$            ;YES!!
11036  054012  122737  000037  001406          CMPB    #31.,RMDA0      ;IS SECTOR > 31??
11037  054020  103412                          BLO     110$            ;YES!!
11038  054022  122737  000035  001406          CMPB    #29.,RMDA0      ;IS SECTOR > 29??
11039  054030  103004                          BHIS    105$            ;NO - IAE SHOULD BE ZERO
11040  054032  032737  010000  001432          BIT     #FMT16,RMOF0    ;18 BIT FORMAT??
11041  054040  001402                          BEQ     110$            ;YES!!
11042  054042  005037  001140          105$:   CLR     $GDDAT          ;IAE SHOULD BE ZERO
11043  054046  013737  001344  001142  110$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
11044  054054  042737  175777  001142          BIC     #^CIAE,$BDDAT
11045  054062  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "IAE" STATUS OK??
11046  054070  001004                          BNE     115$            ;NO!!
11047  054072  042737  040000  056412          BIC     #SKI,500$       ;IAE OK - SKI SHOULD BE 0
11048  054100  000412                          BR      120$
11049  054102  062716  000004          115$:   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11050  054106  112776  000305  000000          MOVB    #305,@(SP)      ;WRITE ERROR NUMBER
11051  054114  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11052  054120  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11053  054122  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11054  054126                          120$:
11055
11056                                  ;REPORT AN ERROR IF "SKI" IS SET AND "IAE" STATUS WAS OK
11057  054126  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11058  054134  042737  137777  001142          BIC     #^CSKI,$BDDAT
11059  054142  013737  056412  001140          MOV     500$,$GDDAT     ;EXPECTED STATUS
11060  054150  042737  137777  001140          BIC     #^CSKI,$GDDAT
11061  054156  032737  040000  001372          BIT     #SKI,RMER2I     ;WAS "SKI" SET??
11062  054164  001417                          BEQ     140$            ;NO!!
11063  054166  032737  040000  056412          BIT     #SKI,500$       ;WAS SKI CAUSED BY IAE = 0??
11064  054174  001032                          BNE     150$            ;YES - DON'T REPORT SKI
11065  054176  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11066  054202  112776  000306  000000          MOVB    #306,@(SP)      ;WRITE ERROR NUMBER
11067  054210  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11068  054214  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11069  054216  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11070  054222  000417                          BR      150$
11071
11072  054224                          140$:
11073
11074                                  ;REPORT AN ERROR IF SKI = 0 AND IAE WAS NOT DETECTED
11075  054224  032737  040000  056412          BIT     #SKI,500$       ;SHOULD SKI BE SET??
11076  054232  001413                          BEQ     150$            ;NO!!
11077  054234  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11078  054240  112776  000307  000000          MOVB    #307,@(SP)      ;WRITE ERROR NUMBER IN CALL
11079  054246  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11080  054252  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11081  054254  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
```

```
11082  054260  000240                          NOP
11083  054262                          150$:
11084
11085                                  ;LOOK FOR "LSC" OR "LBC" OR "DVC" IN ERROR REGISTER #2
11086  054262  032737  006200  001372          BIT     #LSC!LBC!DVC,RMER2I
11087  054270  001512                          BEQ     180$            ;NO ERRORS SET
11088
11089                                  ;REPORT ANY DEVICE FAULT, I.E., "DVC" = 1
11090  054272  032737  000200  001372          BIT     #DVC,RMER2I     ;IS "DVC" = 1??
11091  054300  001424                          BEQ     160$            ;NO!!
11092  054302  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11093  054310  042737  000200  001140          BIC     #DVC,$GDDAT
11094  054316  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11095  054324  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
11096  054330  112776  000310  000000          MOVB    #310,@(SP)      ;WRITE ERROR NUMBER IN CALL
11097  054336  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11098  054342  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11099  054344  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11100  054350  000240                          NOP
11101  054352                          160$:
11102
11103                                  ;REPORT LOSS OF BIT CLOCK, I.E.; "LBC" = 1, IF "MOL" = 1
11104  054352  032737  002000  001372          BIT     #LBC,RMER2I     ;IS LBC SET??
11105  054360  001430                          BEQ     170$            ;NO!!
11106  054362  032737  010000  001342          BIT     #MOL,RMDSI      ;WAS LBC ERROR BY MOL = 0
11107  054370  001424                          BEQ     170$            ;YES!!
11108  054372  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11109  054400  042737  002000  001140          BIC     #LBC,$GDDAT
11110  054406  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11111  054414  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11112  054420  112776  000311  000000          MOVB    #311,@(SP)      ;WRITE ERROR NUMBER IN CALL
11113  054426  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11114  054432  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11115  054434  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11116  054440  000240                          NOP
11117  054442                          170$:
11118
11119                                  ;REPORT LOS OF SYSTEM CLOCK, I.E., "LSC" = 1
11120  054442  032737  004000  001372          BIT     #LSC,RMER2I     ;IS "LSC" = 1??
11121  054450  001422                          BEQ     180$            ;NO!!
11122  054452  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11123  054460  042737  004000  001140          BIC     #LSC,$GDDAT
11124  054466  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11125  054474  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11126  054500  112776  000312  000000          MOVB    #312,@(SP)      ;WRITE ERROR NUMBER
11127  054506  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11128  054510  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11129  054514  000240                          NOP
11130  054516                          180$:
11131
11132                                  ;LOOK FOR "UNS" OR "DTE" OR "WLE" IN ERROR REGISTER #1
11133  054516  032737  054000  001344          BIT     #UNS!DTE!WLE,RMER1I
11134  054524  001527                          BEQ     220$            ;NO BITS SET
11135                                  ;REPORT "UNS" ERROR IF "DVC" = 0
11136  054526  032737  040000  001344          BIT     #UNS,RMER1I     ;IS "UNS" SET??
11137  054534  001427                          BEQ     190$            ;NO!!
```

```
11138  054536  032737  000200  001372          BIT     #DVC,RMER2I     ;WAS "UNS" CAUSED BY "DVC"??
11139  054544  001023                          BNE     190$            ;YES!!
11140  054546  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11141  054554  042737  040000  001140          BIC     #UNS,$GDDAT
11142  054562  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11143  054570  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11144  054574  112776  000313  000000          MOVB    #313,@(SP)      ;WRITE ERROR NUMBER
11145  054602  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11146  054606  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11147  054610  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11148  054614                          190$:
11149
11150                                  ;REPORT ANY DRIVE TIMING ERROR, I.E., "DTE" = 1
11151  054614  032737  010000  001344          BIT     #DTE,RMER1I     ;IS DTE SET??
11152  054622  001423                          BEQ     200$            ;NO!!
11153  054624  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11154  054632  042737  010000  001140          BIC     #DTE,$GDDAT
11155  054640  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11156  054646  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11157  054652  112776  000314  000000          MOVB    #314,@(SP)      ;WRITE ERROR NUMBER IN CALL
11158  054660  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11159  054664  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11160  054666  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11161  054672                          200$:
11162
11163                                  ;REPORT AN ERROR IF WRITE LOCK ERROR IS SET.  SEE IF DRIVE IS NOT
11164                                  ;WRITE PROTECTED, OR IF FUNCTION WAS NOT A WRITE
11165  054672  032737  004000  001344          BIT     #WLE,RMER1I     ;WAS "WLE" SET??
11166  054700  001441                          BEQ     220$            ;NO!!
11167  054702  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11168  054710  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11169  054716  052737  004000  001140          BIS     #WLE,$GDDAT
11170  054724  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11171  054730  112776  000315  000000          MOVB    #315,@(SP)      ;WRITE ERROR NUMBER IN CALL
11172  054736  032737  004000  001342          BIT     #WRL,RMDS1      ;WAS DRIVE WRITE PROTECTED??
11173  054744  001404                          BEQ     205$            ;NO!!
11174  054746  032737  000010  001400          BIT     #BIT3,RMCS10    ;WAS COMMAND A WRITE??
11175  054754  001406                          BEQ     210$            ;YES!!
11176  054756  112776  000316  000000  205$:   MOVB    #316,@(SP)      ;CHANGE ERROR NUMBER
11177  054764  042737  004000  001140          BIC     #WLE,$GDDAT
11178  054772  162716  000002          210$:   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11179  054776  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11180  055000  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11181
11182  055004                          220$:
11183
11184                                  ;OMIT DATA ERROR CHECKS IF ANY PREVIOUS ERRORS HAVE BEEN DETECTED
11185  055004  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
11186  055010  105776  000000                  TSTB    @(SP)           ;WAS ERROR DETECTED??
11187  055014  001404                          BEQ     225$            ;NO - DO DATA CHECKS
11188  055016  162716  000004                  SUB     #4,(SP)         ;RESTORE SP
11189  055022  000137  056024                  JMP     340$            ;SKIP DATA CHECKS
11190  055026  162716  000004          225$:   SUB     #4,(SP)         ;RESTORE SP
11191
11192                                  ;CHECK HEADER ERRORS IF FUNCTION WAS NOT WRITE HEADER AND DATA, AND
11193                                  ;IF HEADER COMPARE IS NOT INHIBITED
```

```
11194  055032  013737  001400  056414          MOV     RMCS10,510$     ;STRIP AND STORE FUNCTION CODE
11195  055040  042737  177700  056414          BIC     #^CFNCMSK,510$
11196  055046  022737  000063  056414          CMP     #WH!GO,510$     ;WAS FUNCTION WRITE HEADER & DATA??
11197  055054  001512                          BEQ     250$            ;YES - SKIP HEADER CHECKS
11198  055056  032737  002000  001362          BIT     #HCI,RMOFI      ;WAS HCI SET??
11199  055064  001106                          BNE     250$            ;YES - SKIP HEADER CHECKS
11200
11201                                  ;SEE IF ANY HEADER ERRORS ARE SET, I.E., "FER" OR "HCRC" OR "HCE"
11202  055066  032737  000620  001344          BIT     #HCRC!FER!HCE,RMER1I
11203  055074  001533                          BEQ     270$            ;NO ERRORS SET
11204
11205                                  ;REPORT HEADER CRC ERROR IF SET
11206  055076  032737  000400  001344          BIT     #HCRC,RMER1I    ;WAS HCRC SET??
11207  055104  001422                          BEQ     230$            ;NO!!
11208  055106  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11209  055114  042737  000400  001140          BIC     #HCRC,$GDDAT
11210  055122  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11211  055130  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
11212  055134  112776  000317  000000          MOVB    #317,@(SP)      ;WRITE ERROR NUMBER
11213  055142  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11214  055146  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11215  055150  000501                          BR      260$
11216  055152                          230$:
11217
11218                                  ;REPORT FORMAT ERROR IF SET
11219  055152  032737  000020  001344          BIT     #FER,RMER1I     ;WAS "FER" SET??
11220  055160  001422                          BEQ     240$            ;NO!!
11221  055162  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11222  055170  042737  000020  001140          BIC     #FER,$GDDAT
11223  055176  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11224  055204  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
11225  055210  112776  000320  000000          MOVB    #320,@(SP)      ;WRITE ERROR NUMBER
11226  055216  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11227  055222  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11228  055224  000453                          BR      260$
11229  055226                          240$:
11230
11231                                  ;REPORT HEADER COMPARE ERROR IF SET
11232  055226  032737  000200  001344          BIT     #HCE,RMER1I     ;WAS "HCE" SET??
11233  055234  001453                          BEQ     270$            ;NO!!
11234  055236  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11235  055244  042737  000200  001140          BIC     #HCE,$GDDAT
11236  055252  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11237  055260  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
11238  055264  112776  000321  000000          MOVB    #321,@(SP)      ;WRITE ERROR NUMBER
11239  055272  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11240  055276  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11241  055300  000425                          BR      260$
11242                                  ;THERE SHOULD BE NO HEADER ERRORS BECAUSE
11243                                  ;       .COMMAND WAS WRITE HEADER AND DATA, OR
11244                                  ;       .HEADER COMPARE INHIBIT WAS SET
11245  055302  032737  000620  001344  250$:   BIT     #HCE!FER!HCRC,RMER1I
11246  055310  001425                          BEQ     270$            ;NO ERRORS WERE SET
11247  055312  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11248  055320  042737  000620  001140          BIC     #HCE!FER!HCRC,$GDDAT
11249  055326  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
```

```
11250  055334  062716  000004                 ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
11251  055340  112776  000322  000000         MOVB    #322,@(SP)       ;WRITE ERROR NUMBER
11252  055346  162716  000002                 SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
11253  055352  004736                         JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
11254  055354  162716  000010         260$:   SUB     #10,(SP)         ;MOVE SP TO NO ERROR
11255  055360  000137  056024                 JMP     340$             ;OMIT FURTHER DATA CHECKS
11256
11257  055364                         270$:
11258
11259                                 ;IF COMMAND WAS A WRITE COMMAND, GO DO WRITE ERROR CHECKS, OTHERWISE
11260                                 ;DO READ ERROR CHECKS
11261  055364  032737  000010  056414         BIT     #BIT3,510$       ;WAS THIS A WRITE COMMAND?
11262  055372  001002                         BNE     275$             ;NO!!
11263  055374  000137  055612                 JMP     310$             ;GO DO WRITE STATUS CHECK
11264  055400                         275$:
11265
11266                                 ;REPORT DATA CHECK IF SET
11267  055400  032737  100000  001344         BIT     #DCK,RMER1I      ;DATA CHECK ERROR??
11268  055406  001450                         BEQ     290$             ;NO!!
11269  055410  013737  001344  001140         MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
11270  055416  042737  100000  001140         BIC     #DCK,$GDDAT
11271  055424  013737  001344  001142         MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
11272  055432  062716  000004                 ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR
11273  055436  112776  000323  000000         MOVB    #323,@(SP)       ;WRITE ERROR NUMBER
11274  055444  032737  004000  001362         BIT     #ECI,RMOFI       ;WAS ECC CORRECTION DISABLED??
11275  055452  001021                         BNE     280$             ;YES!!
11276  055454  112776  000324  000000         MOVB    #324,@(SP)       ;CHANGE TO RECOVERABLE ERROR
11277  055462  032737  000100  001344         BIT     #ECH,RMER1I      ;IS ERROR RECOVERABLE??
11278  055470  001007                         BNE     276$             ;NO !!
11279                                 ;DO NOT REPORT RECOVERABLE ERROR IF READ COMMAND
11280  055472  032737  000020  056414         BIT     #BIT4,510$       ;WAS THIS A READ COMMAND ??
11281  055500  001406                         BEQ     280$             ;NO !!
11282  055502  162716  000004                 SUB     #4,(SP)          ;RESTORE SP
11283  055506  000410                         BR      290$             ;SKIP ERROR - DATA WILL BE CORRECTED
11284  055510  112776  000325  000000 276$:   MOVB    #325,@(SP)       ;CHANGE TO NON RECOVERABLE
11285  055516  162716  000002         280$:   SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
11286  055522  004736                         JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
11287  055524  162716  000010                 SUB     #10,(SP)         ;RESTORE SP TO NO ERROR
11288
11289  055530                         290$:
11290
11291                                 ;REPORT DATA BUS PARITY ERROR IF SET, I.E., MDPE = 1
11292  055530  032737  000400  001340         BIT     #MDPE,RMCS2I     ;PARITY ERROR SET??
11293  055536  001423                         BEQ     300$             ;NO!!
11294  055540  013737  001340  001140         MOV     RMCS2I,$GDDAT    ;EXPECTED STATUS
11295  055546  042737  000400  001140         BIC     #MDPE,$GDDAT
11296  055554  013737  001340  001142         MOV     RMCS2I,$BDDAT    ;RECEIVED STATUS
11297  055562  062716  000004                 ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR
11298  055566  112776  000326  000000         MOVB    #326,@(SP)       ;WRITE ERROR NUMBER
11299  055574  162716  000002                 SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
11300  055600  004736                         JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
11301  055602  162716  000010                 SUB     #10,(SP)         ;MOVE SP TO NO ERROR
11302  055606  000137  056024         300$:   JMP     340$             ;SKIP WRITE STATUS CHECK
11303
11304  055612                         310$:
11305
```

```
11306
11307  055612  032737  000001  001342          ;TEST TO SEE THAT OFFSET MODE WAS RESET; REPORT ERROR IF "OM" = 1
11308  055620  001423                                  BIT     #OM,RMDS1       ;IS OFFSET ON??
11309  055622  013737  001342  001140                  BEQ     320$            ;NO
11310  055630  042737  000001  001140                  MOV     RMDS1,$GDDAT    ;EXPECTED STATUS
11311  055636  013737  001342  001142                  BIC     #OM,$GDDAT
11312  055644  062716  000004                          MOV     RMDS1,$BDDAT    ;RECEIVED STATUS
11313  055650  112776  000327  000000                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11314  055656  162716  000002                          MOVB    #327,@(SP)      ;WRITE ERROR NUMBER IN CALL
11315  055662  004736                                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11316  055664  162716  000010                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11317  055670                                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11318                                          320$:
11319                                          ;TEST FOR DATA BUS PARITY ERROR; REPORT ERROR IF "DPE" = 1
11320  055670  032737  000010  001372                  BIT     #DPE,RMER21     ;DATA PARITY ERROR??
11321  055676  001423                                  BEQ     330$            ;NO!!
11322  055700  013737  001372  001140                  MOV     RMER21,$GDDAT   ;EXPECTED STATUS
11323  055706  042737  000010  001140                  BIC     #DPE,$GDDAT
11324  055714  013737  001372  001142                  MOV     RMER21,$BDDAT   ;RECEIVED STATUS
11325  055722  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11326  055726  112776  000330  000000                  MOVB    #330,@(SP)      ;WRITE ERROR NUMBER
11327  055734  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11328  055740  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11329  055742  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11330  055746                                  330$:
11331
11332                                          ;TEST FOR WRITE CLOCK FAILURE; REPORT ERROR IF "WCF" = 1
11333  055746  032737  000040  001344                  BIT     #WCF,RMER11     ;IS "WCF" SET??
11334  055754  001423                                  BEQ     340$            ;NO!!
11335  055756  013737  001344  001140                  MOV     RMER11,$GDDAT   ;EXPECTED STATUS
11336  055764  042737  000040  001140                  BIC     #WCF,$GDDAT
11337  055772  013737  001344  001142                  MOV     RMER11,$BDDAT   ;RECEIVED STATUS
11338  056000  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11339  056004  112776  000331  000000                  MOVB    #331,@(SP)      ;WRITE ERROR NUMBER
11340  056012  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11341  056016  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11342  056020  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11343  056024                                  340$:
11344
11345                                          ;REPORT "DATA LATE" ERROR IF "DLT" = 1
11346  056024  032737  100000  001340                  BIT     #DLT,RMCS21     ;IS "DLT" SET??
11347  056032  001423                                  BEQ     350$            ;NO!!
11348  056034  013737  001340  001140                  MOV     RMCS21,$GDDAT   ;EXPECTED STATUS
11349  056042  042737  100000  001140                  BIC     #DLT,$GDDAT
11350  056050  013737  001340  001142                  MOV     RMCS21,$BDDAT   ;RECEIVED STATUS
11351  056056  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11352  056062  112776  000332  000000                  MOVB    #332,@(SP)      ;WRITE ERROR NUMBER
11353  056070  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11354  056074  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11355  056076  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11356  056102                                  350$:
11357                                          ;LOOK FOR UNEXPECTED CHANGES IN DRIVE STATUS
11358  056102  013746  001342                  MOV     RMDS1,-(SP)     ;STACK DRIVE STATUS
11359  056106  042716  147677                  BIC     #^C<PIP!MOL!VV>,(SP) ;CLEAR DONT CARES
11360  056112  022726  010100                  CMP     #MOL!VV,(SP)+   ;IS DRIVE STATUS OK??
11361  056116  001522                          BEQ     380$            ;YES!!
```

```
11362
11363                                   ;REPORT ERROR IF POSITIONING IN PROGRESS AND NO SEEK INCOMPLETE ERROR,
11364                                   ;I.E, PIP = 1 AND SKI = 0
11365  056120  032737  020000  001342          BIT     #PIP,RMDSI      ;IS "PIP" SET??
11366  056126  001430                          BEQ     360$            ;NO!!
11367  056130  032737  040000  001372          BIT     #SKI,RMER2I     ;WAS "SKI" ERROR REPORTED??
11368  056136  001024                          BNE     360$            ;YES-DONT REPORT PIP
11369  056140  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11370  056146  042737  020000  001140          BIC     #PIP,$GDDAT
11371  056154  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11372  056162  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11373  056166  112776  000333  000000          MOVB    #333,@(SP)      ;WRITE ERROR NUMBER
11374  056174  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11375  056200  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11376  056202  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11377  056206  000240                          NOP
11378  056210                          360$:
11379
11380                                   ;REPORT ERROR IF MEDIUM IS NOT ON LINE AND OPI ERROR WAS NOT
11381                                   ;REPORTED, I.E., MOL = OPI = 0
11382  056210  032737  010000  001342          BIT     #MOL,RMDSI      ;IS MEDIUM ON LINE??
11383  056216  001027                          BNE     370$            ;YES!!
11384  056220  032737  020000  001344          BIT     #OPI,RMER1I     ;WAS OPI ERROR REPORTED??
11385  056226  001023                          BNE     370$            ;YES!!
11386  056230  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11387  056236  052737  010000  001140          BIS     #MOL,$GDDAT
11388  056244  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11389  056252  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
11390  056256  112776  000334  000000          MOVB    #334,@(SP)      ;WRITE ERROR NUMBER
11391  056264  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11392  056270  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11393  056272  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11394  056276                          370$:
11395
11396                                   ;REPORT ERROR IF VOLUME IS NOT VALID AND "IVC" ERROR WAS NOT
11397                                   ;REPORTED, I.E., VV = IVC = 0
11398  056276  032737  000100  001342          BIT     #VV,RMDSI       ;IS VOLUME VALID??
11399  056304  001027                          BNE     380$            ;YES!!
11400  056306  032737  010000  001372          BIT     #IVC,RMER2I     ;WAS IVC ERROR REPORTED??
11401  056314  001033                          BNE     390$            ;YES!!
11402  056316  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11403  056324  052737  000100  001140          BIS     #VV,$GDDAT
11404  056332  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11405  056340  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11406  056344  112776  000335  000000          MOVB    #335,@(SP)      ;WRITE ERROR NUMBER
11407  056352  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11408  056356  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11409  056360  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11410  056364                          380$:
11411
11412                                   ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS FOUND
11413  056364  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11414  056370  105776  000000                  TSTB    @(SP)           ;ANY ERROR??
11415  056374  001403                          BEQ     390$            ;NO!!
11416  056376  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
11417  056402  000402                          BR      400$
```

```
11418  056404  162716  000004          390$:   SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11419
11420  056410  000207                  400$:   RTS     PC              ;RETURN TO USER
11421
11422  056412  000000                  500$:   .WORD                   ;ERROR FLAGS
11423  056414  000000                  510$:   .WORD                   ;TEMPORARY STORAGE
```

```
11424                                   .SBTTL  STATIC DRIVE STATUS CHECK SUBROUTINE
11425
11426                                   ;THIS SUBROUTINE LOOKS FOR UNEXPECTED CHANGES IN DRIVE
11427                                   ;STATUS, SUCH AS THE DRIVE LOSING VOLUME VALID.  THE SUBROUTINE
11428                                   ;CAN BE USED BY HOUSEKEEPING AND OTHER COMMANDS DURING WHICH THERE
11429                                   ;SHOULD NOT BE ANY DRIVE ERRORS OR CHANGES IN STATE.
11430
11431                                   ;THE FOLLOWING CONDITIONS ARE TESTED AND REPORTED AS ERRORS
11432                                   ;IF TRUE:
11433
11434                                   ;       .MOL = 0, INDICATES DRIVE WENT OFFLINE, NOTE
11435                                   ;THAT MOL IS ASSUMED TO HAVE BEEN SET
11436                                   ;       .VV = 0, INDICATES THE DRIVE LOST VOLUME VALID
11437                                   ;       .PIP = 1, INDICATES THAT THE DRIVE IS OFF CYLINDER
11438                                   ;       .SKI = 1, INDICATES THE DRIVE HAS AN UNEXPECTED SKI ERROR
11439                                   ;       .DVC = 1, INDICATES AN UNEXPECTED DEVICE FAULT
11440
11441                                   ;THE SUBROUTINE IS CALLED AFTER STORING STATUS IN THE GET BUFFER.
11442
11443                                   ;(1)    JSR     PC,STCDRVSTS
11444                                   ;       BR      ???             RETURN HERE IF NO ERROR
11445                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
11446                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11447                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11448                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
11449
11450  056416                           STCDRVSTS:
11451
11452                                   ;CLEAR USER'S ERROR CALL
11453  056416  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11454  056422  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
11455  056426  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR RETURN
11456                                   ;SEE IF "MOL" = "VV" = 1, AND "PIP" = 0
11457  056432  013746  001342                  MOV     RMDSI,-(SP)     ;PUT DRIVE STATUS ON STACK
11458  056436  042716  147677                  BIC     #^C<PIP!MOL!VV>,(SP)
11459  056442  022726  010100                  CMP     #MOL!VV,(SP)+   ;ARE MOL,VV AND PIP O.K.??
11460  056446  001524                          BEQ     30$             ;YES!!
11461
11462                                   ;REPORT AN ERROR IF MOL = 0 AND "OPI" = 0
11463  056450  032737  010000  001342          BIT     #MOL,RMDSI      ;IS MOL ON ??
11464  056456  001030                          BNE     10$             ;YES!!
11465  056460  032737  020000  001344          BIT     #OPI,RMER11     ;WAS "OPI" SET??
11466  056466  001024                          BNE     10$             ;YES-DONT REPORT "MOL" = 0
11467  056470  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11468  056476  052737  010000  001140          BIS     #MOL,$GDDAT
11469  056504  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11470  056512  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11471  056516  112776  000207  000000          MOVB    #207,@(SP)      ;WRITE ERROR NUMBER IN CALL
11472  056524  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11473  056530  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11474  056532  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
```

```
11475  056536  000240                          NOP
11476  056540                          10$:
11477
11478                                  ;REPORT AN ERROR IF VOLUME VALID IS NOW ZERO AND "IVC" = 0
11479  056540  032737  000100  001342          BIT     #VV,RMDS1       ;IS "VV" = 0??
11480  056546  001030                          BNE     20$             ;NO!!
11481  056550  032737  010000  001372          BIT     #IVC,RMER21     ;WAS "IVC" SET??
11482  056556  001024                          BNE     20$             ;YES-DONT REPORT "VV" = 0
11483  056560  013737  001342  001140          MOV     RMDS1,$GDDAT    ;EXPECTED STATUS
11484  056566  052737  000100  001342          BIS     #VV,RMDS1
11485  056574  013737  001342  001142          MOV     RMDS1,$BDDAT    ;RECEIVED STATUS
11486  056602  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11487  056606  112776  000210  000000          MOVB    #210,@(SP)      ;WRITE ERROR NUMBER IN CALL
11488  056614  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11489  056620  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11490  056622  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11491  056626  000240                          NOP
11492  056630                          20$:
11493
11494                                  ;REPORT AN ERROR IF DRIVE IS OFF CYLINDER AND "SKI" = 0
11495  056630  032737  020000  001342          BIT     #PIP,RMDS1      ;IS DRIVE OFF CYLINDER??
11496  056636  001430                          BEQ     30$             ;NO!!
11497  056640  032737  040000  001372          BIT     #SKI,RMER21     ;WAS "SKI" SET??
11498  056646  001024                          BNE     30$             ;YES-DONT REPORT "PIP" = 1
11499  056650  013737  001342  001140          MOV     RMDS1,$GDDAT    ;EXPECTED STATUS
11500  056656  042737  020000  001140          BIC     #PIP,$GDDAT
11501  056664  013737  001342  001142          MOV     RMDS1,$BDDAT    ;RECEIVED STATUS
11502  056672  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11503  056676  112776  000211  000000          MOVB    #211,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
11504  056704  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11505  056710  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11506  056712  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11507  056716  000240                          NOP
11508  056720                          30$:
11509
11510                                  ;SEE IF "SKI" = "DVC" = 0
11511  056720  013746  001372                  MOV     RMER21,-(SP)    ;PUT ERROR REG 2 ON STACK
11512  056724  042726  137577                  BIC     #^C<SKI!DVC>,(SP)+
11513  056730  001460                          BEQ     60$             ;BRANCH IF NO ERROR
11514  056732                          40$:
11515
11516                                  ;REPORT AN ERROR IF THERE IS A DEVICE FAULT
11517  056732  032737  000200  001372          BIT     #DVC,RMER21     ;ANY DEVICE FAULT??
11518  056740  001424                          BEQ     50$             ;NO!!
11519  056742  013737  001372  001140          MOV     RMER21,$GDDAT   ;EXPECTED STATUS
11520  056750  042737  000200  001140          BIC     #DVC,$GDDAT
11521  056756  013737  001372  001142          MOV     RMER21,$BDDAT   ;RECEIVED STATUS
11522  056764  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USEP'S CALL
11523  056770  112776  000212  000000          MOVB    #212,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11524  056776  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11525  057002  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11526  057004  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11527  057010  000240                          NOP
11528  057012                          50$:
11529
11530                                  ;REPORT AN ERROR IF "SKI" = 1
```

```
11531  057012  032737  040000  001372          BIT     #SKI,RMER21     ;IS THERE A SEEK INCOMPLETE ERROR
11532  057020  001424                          BEQ     60$             ;NO!!
11533  057022  013737  001372  001140          MOV     RMER21,$GDDAT   ;EXPECTED STATUS
11534  057030  042737  040000  001140          BIC     #SKI,$GDDAT
11535  057036  013737  001372  001142          MOV     RMER21,$BDDAT   ;RECEIVED STATUS
11536  057044  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11537  057050  112776  000213  000000          MOVB    #213,@(SP)      ;WRITE ERROR NUMBER IN USER'S ERROR CALL
11538  057056  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11539  057062  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11540  057064  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11541  057070  000240                          NOP
11542  057072                          60$:
11543
11544                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
11545  057072  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11546  057076  105776  000000                  TSTB    @(SP)           ;WAS AN ERROR DETECTED??
11547  057102  001403                          BEQ     70$             ;NO!!
11548  057104  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO USER'S ERROR RETURN
11549  057110  000402                          BR      80$
11550  057112  162716  000004          70$:    SUB     #4,(SP)         ;NO - MOVE SP TO NO ERROR RETURN
11551  057116  000240                  80$:    NOP
11552  057120  000207                          RTS     PC              ;RETURN TO USER
```

```
11553                                   .SBTTL  COMPOSITE ERROR CHECK SUBROUTINE
11554
11555                                   ;THIS SUBROUTINE CHECKS THE STORED CONTENTS OF RMER1 AND
11556                                   ;RMER2 AFTER MASKING EACH REGISTER WORD WITH THE USER'S STATUS
11557                                   ;MASKS AND REPORTS AN ERROR IF ANY BITS ARE LEFT ON AFTER
11558                                   ;THE MASKS ARE APPLIED.
11559
11560                                   ;CALL:
11561                                   ;(1)    JSR     PC,CMPERRSTS
11562                                   ;       .WORD                   MASK FOR ERROR REGISTER 1
11563                                   ;       .WORD                   MASK FOR ERROR REGISTER 2
11564                                   ;       BR      ???             RETURN HERE IF NO ERROR
11565                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
11566                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11567                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11568                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
11569
11570                                   ;NOTE:  BITS TO BE MASKED SHOULD BE ONE; BITS TO BE TESTED SHOULD
11571                                   ;BE ZERO
11572
11573  057122                           CMPERRSTS:
11574
11575                                   ;MASK AND STORE THE CONTENTS OF RMER1  AND RMER2
11576  057122  013737  001344  001176           MOV     RMER1I,$TMP1    ;STORE RMER1 AT TEMP STORAGE
11577  057130  047637  000000  001176           BIC     @(SP),$TMP1     ;MASK   RMER1
11578  057136  062716  000002                   ADD     #2,(SP)         ;MOVE SP TO NEXT MASK
11579  057142  013737  001372  001200           MOV     RMER2I,$TMP2    ;STORE RMER2 AT TEMP STORAGE
11580  057150  047637  000000  001200           BIC     @(SP),$TMP2     ;MASK RMER2
11581
11582
11583                                   ;CLEAR USER'S ERROR CALL
11584  057156  062716  000006                   ADD     #6,(SP)         ;MOVE SP TO USER'S ERROR CALL
11585  057162  105076  000000                   CLRB    @(SP)           ;CLEAR ERROR NUMBER
11586  057166  162716  000004                   SUB     #4,(SP)         ;LEAVE SP AT NO ERROR RETURN
11587
11588                                   ;SEE IF THERE WERE ANY ERRORS IN RMER1, I.E., $TMP1
11589  057172  005737  001176                   TST     $TMP1           ;ANY ERRORS TO REPORT??
11590  057176  001420                           BEQ     5$              ;NO !!
11591  057200  013737  001176  001142           MOV     $TMP1,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
11592  057206  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
11593  057212  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11594  057216  112776  000066  000000           MOVB    #66,@(SP)       ;CORRECTABLE DATA CHECK ERROR #
11595  057224  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11596  057230  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11597  057232  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11598  057236  000240                           NOP
11599  057240                           5$:
11600
11601
11602                                   ;SEE IF THERE ARE ANY ERRORS TO REPORT IN RMER2 ($TMP2)
11603  057240  005737  001200                   TST     $TMP2           ;ANY ERRORS IN RMER2?
11604  057244  001420                           BEQ     10$             ;NO!!
11605
11606  057246  013737  001200  001142           MOV     $TMP2,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
11607  057254  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
11608  057260  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
```

```
11609  057264  112776  000067  000000          MOVB    #67,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11610  057272  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11611  057276  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11612  057300  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11613  057304  000240                          NOP
11614  057306                          10$:
11615
11616                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
11617  057306  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11618  057312  105776  000000                  TSTB    @(SP)           ;WAS THERE AN ERROR CALLED??
11619  057316  001403                          BEQ     20$             ;NO!!
11620  057320  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
11621  057324  000402                          BR      30$
11622  057326  162716  000004          20$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11623  057332  000207                  30$:    RTS     PC              ;RETURN TO USER
11624
11625
```

```
11626                                   .SBTTL  STOP AND SHUTDOWN SUBROUTINES
11627
11628  057334                           STOP:
11629
11630                                   ;DROP PRIORITY TO ALLOW CONSOLE INTERRUPT
11631  057334  012746  000140                   MOV     #PR3,-(SP)      ;;PUT NEW PS ON STACK
11632  057340  012746  057346                   MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
11633  057344  000002                           RTI                     ;;POP NEW PC AND PS
11634  057346                           64$:
11635  057346  000240                           NOP
11636                                   ;RAISE PRIORITY TO INHIBIT INTERRUPT
11637  057350  012746  000300                   MOV     #PR6,-(SP)      ;;PUT NEW PS ON STACK
11638  057354  012746  057362                   MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
11639  057360  000002                           RTI                     ;;POP NEW PC AND PS
11640  057362                           65$:
11641
11642  057362  000207                   10$:    RTS     PC              ;CONTINUE
11643
11644
11645  057364  012737  177777  001326   SHUT2:  MOV     #-1,CTLFG       ;SET CONTROL-C FLAG             ;;++C
11646  057372                           SHUT:
11647  057372  005737  001326                   TST     @#CTLFG         ;HALT ?
11648  057376  001002                           BNE     .+6             ;BRANHC IF YES
11649  057400  000137  007124                   JMP     READY           ;CONTINUE
11650  057404  104401  057426                   TYPE    ,100$           ;TYPE THE HALT MESSAGE
11651  057410  005737  000042                   TST     42              ;RUNNING STANDALONE ??
11652  057414  001402                           BEQ     10$             ;YES !!
11653  057416  000137  032766                   JMP     $EOP            ;NO - GO TO END OF PASS
11654  057422                           10$:
11655  057422  000137  005324                   JMP     START           ;GO TO START
11656  057426  042524  052123  053440  100$:   .ASCIZ  @TEST WAS HALTED@<CR><LF><LF>
11657  057434  051501  044040  046101
11658  057442  042524  006504  005012
11659  057450     000
11660          057452                   .EVEN
```

```
11661                                   .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
11662
11663                                   ;;**********************************************************************
11664                                   ;*SAVE R0-R5
11665                                   ;*CALL:
11666                                   ;*      SAVREG
11667                                   ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
11668                                   ;*
11669                                   ;*TOP---(+16)
11670                                   ;* +2---(+18)
11671                                   ;* +4---R5
11672                                   ;* +6---R4
11673                                   ;* +8---R3
11674                                   ;*+10---R2
11675                                   ;*+12---R1
11676                                   ;*+14---R0
11677
11678  057452                           $SAVREG:
11679  057452  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11680  057454  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11681  057456  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11682  057460  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
11683  057462  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
11684  057464  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
11685  057466  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
11686  057472  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
11687  057476  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
11688  057502  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
11689  057506  000002                          RTI
11690
11691                                   ;*RESTORE R0-R5
11692                                   ;*CALL:
11693                                   ;*      RESREG
11694  057510                           $RESREG:
11695  057510  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
11696  057514  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
11697  057520  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
11698  057524  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
11699  057530  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
11700  057532  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
11701  057534  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
11702  057536  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
11703  057540  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11704  057542  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11705  057544  000002                          RTI
11706                                   .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
11707
11708                                   ;;**********************************************************************
11709                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
11710                                   ;*BINARY-ASCII NUMBER AND TYPE IT.
11711                                   ;*CALL:
11712                                   ;*      MOV     NUMBER,-(SP)    ;;NUMBER TO BE TYPED
11713                                   ;*      TYPBN                   ;;TYPE IT
11714
11715  057546  010146                   $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
11716  057550  016601  000006                  MOV     6(SP),R1        ;;GET THE INPUT NUMBER
```

```
11717  057554  000261                          SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
11718  057556  112737  000060  057620  1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
11719  057564  006101                          ROL     R1              ;;GET THIS BIT
11720  057566  001406                          BEQ     2$              ;;DONE?
11721  057570  105537  057620                  ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
11722  057574  104401  057620                  TYPE    ,$BIN           ;;GO TYPE THIS BIT
11723  057600  000241                          CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
11724  057602  000765                          BR      1$              ;;GO DO THE NEXT BIT
11725  057604  012601                  2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
11726  057606  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
11727  057614  012616                          MOV     (SP)+,(SP)
11728  057616  000002                          RTI                     ;;RETURN TO USER
11729  057620     000     000          $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
11730                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
11731
11732                                  ;;*****************************************************************
11733                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
11734                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
11735                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
11736                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
11737                                  ;*REPLACED WITH SPACES.
11738                                  ;*CALL:
11739                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
11740                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
11741
11742  057622                          $TYPDS:
11743  057622  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11744  057624  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11745  057626  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11746  057630  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
11747  057632  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
11748  057634  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
11749  057640  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
11750  057644  100004                          BPL     1$              ;;BR IF INPUT IS POS.
11751  057646  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
11752  057650  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
11753  057656  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
11754  057660  012703  060036                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
11755  057664  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
11756  057670  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
11757  057672  016001  060026                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
11758  057676  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
11759  057700  002402                          BLT     4$              ;;BR IF DONE
11760  057702  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
11761  057704  000774                          BR      3$
11762  057706  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
11763  057710  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
11764  057712  001002                          BNE     5$              ;;FALL THROUGH IF 0
11765  057714  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
11766  057716  100407                          BMI     7$              ;;BR IF YES
11767  057720  106316                  5$:     ASLB    (SP)            ;;MSD?
11768  057722  103003                          BCC     6$              ;;BR IF NO
11769  057724  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
11770  057732  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
11771  057736  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
11772  057742  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
```

```
11773  057744  005720                          TST     (R0)+           ;;JUST INCREMENTING
11774  057746  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
11775  057752  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
11776  057754  003002                          BGT     8$              ;;GO TO EXIT
11777  057756  010502                          MOV     R5,R2           ;;GET THE LSD
11778  057760  000764                          BR      6$              ;;GO CHANGE TO ASCII
11779  057762  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
11780  057764  100003                          BPL     9$              ;;BR IF NO
11781  057766  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
11782  057774  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
11783  057776  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
11784  060000  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
11785  060002  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
11786  060004  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11787  060006  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11788  060010  104401  060036                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
11789  060014  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
11790  060022  012616                          MOV     (SP)+,(SP)
11791  060024  000002                          RTI                     ;;RETURN TO USER
11792  060026  023420                  $DTBL:  10000.
11793  060030  001750                          1000.
11794  060032  000144                          100.
11795  060034  000012                          10.
11796  060036  000004                  $DBLK:  .BLKW   4
11797                                   .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
11798
11799                                   ;;**************************************************************************
11800                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
11801                                   ;*OCTAL (ASCII) NUMBER AND TYPE IT.
11802                                   ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
11803                                   ;*CALL:
11804                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11805                                   ;*      TYPOS                   ;;CALL FOR TYPEOUT
11806                                   ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
11807                                   ;*      .BYTE   M               ;;M=1 OR 0
11808                                   ;*                                      ;;1=TYPE LEADING ZEROS
11809                                   ;*                                      ;;0=SUPPRESS LEADING ZEROS
11810                                   ;*
11811                                   ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
11812                                   ;*$TYPOS OR $TYPOC
11813                                   ;*CALL:
11814                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11815                                   ;*      TYPON                   ;;CALL FOR TYPEOUT
11816                                   ;*
11817                                   ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
11818                                   ;*CALL:
11819                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11820                                   ;*      TYPOC                   ;;CALL FOR TYPEOUT
11821
11822  060046  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
11823  060052  116637  000001  060271          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
11824  060060  112637  060273                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
11825  060064  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
11826  060070  000406                          BR      $TYPON
11827  060072  112737  000001  060271  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
11828  060100  112737  000006  060273          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
```

```
11829  060106  112737  000005  060270  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
11830  060114  010346                          MOV     R3,-(SP)        ;;SAVE R3
11831  060116  010446                          MOV     R4,-(SP)        ;;SAVE R4
11832  060120  010546                          MOV     R5,-(SP)        ;;SAVE R5
11833  060122  113704  060273                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
11834  060126  005404                          NEG     R4
11835  060130  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
11836  060134  110437  060272                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
11837  060140  113704  060271                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
11838  060144  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
11839  060150  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
11840  060152  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
11841  060154  000404                          BR      3$              ;;GO DO MSB
11842  060156  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
11843  060160  006105                          ROL     R5
11844  060162  006105                          ROL     R5
11845  060164  010503                          MOV     R5,R3
11846  060166  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
11847  060170  105337  060272                  DECB    $OMODE          ;;TYPE THIS DIGIT?
11848  060174  100016                          BPL     7$              ;;BR IF NO
11849  060176  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
11850  060202  001002                          BNE     4$              ;;TEST FOR 0
11851  060204  005704                          TST     R4              ;;SUPPRESS THIS 0?
11852  060206  001403                          BEQ     5$              ;;BR IF YES
11853  060210  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
11854  060212  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
11855  060216  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
11856  060222  110337  060266                  MOVB    R3,8$           ;;SAVE FOR TYPING
11857  060226  104401  060266                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
11858  060232  105337  060270          7$:     DECB    $OCNT           ;;COUNT BY 1
11859  060236  003347                          BGT     2$              ;;BR IF MORE TO DO
11860  060240  002402                          BLT     6$              ;;BR IF DONE
11861  060242  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
11862  060244  000744                          BR      2$              ;;GO DO THE LAST DIGIT
11863  060246  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
11864  060250  012604                          MOV     (SP)+,R4        ;;RESTORE R4
11865  060252  012603                          MOV     (SP)+,R3        ;;RESTORE R3
11866  060254  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
11867  060262  012616                          MOV     (SP)+,(SP)
11868  060264  000002                          RTI                     ;;RETURN
11869  060266     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
11870  060267     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
11871  060270     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
11872  060271     000          $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
11873  060272  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
11874                          .SBTTL  TYPE ROUTINE
11875
11876                          ;;**********************************************************************
11877                          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
11878                          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
11879                          ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
11880                          ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
11881                          ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
11882                          ;*
11883                          ;*CALL:
11884                          ;*1) USING A TRAP INSTRUCTION
```

```
11885                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
11886                                   ;*OR
11887                                   ;*      TYPE
11888                                   ;*      MESADR
11889                                   ;*
11890
11891  060274  105737  001173          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
11892  060300  100002                          BPL     1$              ;;BR IF YES
11893  060302  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
11894  060304  000430                          BR      3$              ;;LEAVE
11895  060306  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
11896  060310  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
11897  060314  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
11898  060322  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
11899  060324  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
11900  060332  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
11901  060334  010037  060344                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
11902  060340  004737  063220                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
11903  060344  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
11904  060346  132737  000040  001243  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
11905  060354  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
11906  060356  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
11907  060360  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
11908  060362  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
11909  060364  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
11910  060366  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
11911  060372  000002                          RTI                     ;;RETURN
11912  060374  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
11913  060400  001430                          BEQ     8$
11914  060402  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
11915  060406  001006                          BNE     5$
11916  060410  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
11917  060412  104401                          TYPE                    ;;TYPE A CR AND LF
11918  060414  001217                          $CRLF
11919  060416  105037  060552                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
11920  060422  000755                          BR      2$              ;;GET NEXT CHARACTER
11921  060424  004737  060506          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
11922  060430  123726  001172          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
11923  060434  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
11924  060436  013746  001170                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
11925                                                                  ;;AND THE NULL CHAR.
11926  060442  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
11927  060446  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
11928  060450  004737  060506                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
11929  060454  105337  060552                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
11930  060460  000770                          BR      7$              ;;LOOP
11931
11932                                   ;HORIZONTAL TAB PROCESSOR
11933
11934  060462  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
11935  060466  004737  060506          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
11936  060472  132737  000007  060552          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
11937  060500  001372                          BNE     9$              ;;TAB STOP
11938  060502  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
11939  060504  000724                          BR      2$              ;;GET NEXT CHARACTER
11940  060506  105777  120452          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
```

```
11941  060512  100375                          BPL     $TYPEC
11942  060514  116677  000002  120444          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
11943  060522  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
11944  060530  001003                          BNE     1$              ;;BRANCH IF NO
11945  060532  105037  060552                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
11946  060536  000406                          BR      $TYPEX          ;;EXIT
11947  060540  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
11948  060546  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
11949  060550  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
11950  060552  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
11951  060554  000207                  $TYPEX: RTS     PC
11952
11953                                   .SBTTL  SCOPE HANDLER ROUTINE
11954
11955                                   ;;**********************************************************************
11956                                   ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
11957                                   ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
11958                                   ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
11959                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11960                                   ;*SW14=1        LOOP ON TEST
11961                                   ;*SW11=1        INHIBIT ITERATIONS
11962                                   ;*SW09=1        LOOP ON ERROR
11963                                   ;*SW08=1        LOOP ON TEST IN SWR<7:0>
11964                                   ;*CALL
11965                                   ;*      SCOPE           ;;SCOPE=IOT
11966
11967  060556                           $SCOPE:
11968  060556  104410                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
11969  060560  004737  057334                  JSR     PC,STOP
11970  060564  032777  040000  120362  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
11971  060572  001131                          BNE     $OVER           ;;YES IF SW14=1
11972                                   ;#####START OF CODE FOR THE XOR TESTER#####
11973  060574  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
11974                                                                   ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
11975  060576  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
11976  060602  012737  060622  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
11977  060610  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
11978  060614  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11979  060620  000500                          BR      $SVLAD          ;;GO TO THE NEXT TEST
11980  060622  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
11981  060624  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11982  060630  000440                          BR      7$              ;;LOOP ON THE PRESENT TEST
11983  060632                          6$:;#####END OF CODE FOR THE XOR TESTER#####
11984  060632  032777  000400  120314          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
11985  060640  001421                          BEQ     2$              ;;BR IF NO
11986  060642  005046                          CLR     -(SP)           ;;CLEAR A TEMP. LOCATION
11987  060644  117716  120304                  MOVB    @SWR,(SP)       ;;PICKUP THE DESIRED TEST NUMBER
11988  060650  001414                          BEQ     8$              ;;BRANCH IF BAD TEST NUMBER IN SWR
11989  060652  022716  000035                  CMP     #35,(SP)        ;;CHECK THE NUMBER IN THE SWR
11990  060656  002411                          BLT     8$              ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
11991  060660  011637  001116                  MOV     (SP),$TSTNM     ;;UPDATE THE TEST NUMBER
11992  060664  005316                          DEC     (SP)            ;;BACKUP BY ONE
11993  060666  006316                          ASL     (SP)            ;;SCALE THE TEST NUMBER AS AN INDEX
11994  060670  062716  061074                  ADD     #$SW08TBL,(SP)  ;;FORM THE ADDRESS OF TEST POINTER
11995  060674  013637  001122                  MOV     @(SP)+,$LPADR   ;;SET LOOP ADDRESS TO DESIRED TEST
11996  060700  000466                          BR      $OVER           ;;GO LOOP ON THE TEST
```

```
11997  060702  005726                  8$:     TST     (SP)+           ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
11998  060704  105737  001117          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
11999  060710  001421                          BEQ     3$              ;;BR IF NO
12000  060712  123737  001131  001117          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
12001  060720  101015                          BHI     3$              ;;BR IF NO
12002  060722  032777  001000  120224          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
12003  060730  001404                          BEQ     4$              ;;BR IF NO
12004  060732  013737  001124  001122  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
12005  060740  000446                          BR      $OVER
12006  060742  105037  001117          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
12007  060746  005037  001206                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
12008  060752  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
12009  060754  032777  004000  120172  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
12010  060762  001011                          BNE     1$              ;;BR IF YES
12011  060764  005737  001230                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
12012  060770  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
12013  060772  005237  001120                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
12014  060776  023737  001206  001120          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
12015  061004  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
12016  061006  012737  000001  001120  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
12017  061014  013737  061072  001206          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
12018  061022  105237  001116          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
12019  061026  113737  001116  001226          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
12020  061034  011637  001122                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
12021  061040  011637  001124                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
12022  061044  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
12023  061050  112737  000001  001131          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
12024  061056  013777  001116  120072  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
12025  061064  013716  001122                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
12026  061070  000002                          RTI                     ;;FIXES PS
12027  061072  000012          $MXCNT: 10.                             ;;MAX. NUMBER OF ITERATIONS
12028  061074          $SW08TBL:
12029  061074  007164                  .WORD   TST1+2          ;;STARTING ADDRESS OF TEST 1
12030  061076  007362                  .WORD   TST2+2          ;;STARTING ADDRESS OF TEST 2
12031  061100  007546                  .WORD   TST3+2          ;;STARTING ADDRESS OF TEST 3
12032  061102  007730                  .WORD   TST4+2          ;;STARTING ADDRESS OF TEST 4
12033  061104  010530                  .WORD   TST5+2          ;;STARTING ADDRESS OF TEST 5
12034  061106  011330                  .WORD   TST6+2          ;;STARTING ADDRESS OF TEST 6
12035  061110  012144                  .WORD   TST7+2          ;;STARTING ADDRESS OF TEST 7
12036  061112  012720                  .WORD   TST10+2         ;;STARTING ADDRESS OF TEST 10
12037  061114  013640                  .WORD   TST11+2         ;;STARTING ADDRESS OF TEST 11
12038  061116  014440                  .WORD   TST12+2         ;;STARTING ADDRESS OF TEST 12
12039  061120  015214                  .WORD   TST13+2         ;;STARTING ADDRESS OF TEST 13
12040  061122  016132                  .WORD   TST14+2         ;;STARTING ADDRESS OF TEST 14
12041  061124  016706                  .WORD   TST15+2         ;;STARTING ADDRESS OF TEST 15
12042  061126  017462                  .WORD   TST16+2         ;;STARTING ADDRESS OF TEST 16
12043  061130  020236                  .WORD   TST17+2         ;;STARTING ADDRESS OF TEST 17
12044  061132  021040                  .WORD   TST20+2         ;;STARTING ADDRESS OF TEST 20
12045  061134  021564                  .WORD   TST21+2         ;;STARTING ADDRESS OF TEST 21
12046  061136  022310                  .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
12047  061140  023032                  .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
12048  061142  023330                  .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
12049  061144  023626                  .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
12050  061146  024354                  .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
12051  061150  025102                  .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
12052  061152  025630                  .WORD   TST30+2         ;;STARTING ADDRESS OF TEST 30
```

```
12053  061154  026522                        .WORD   TST31+2         ;;STARTING ADDRESS OF TEST 31
12054  061156  027330                        .WORD   TST32+2         ;;STARTING ADDRESS OF TEST 32
12055  061160  030052                        .WORD   TST33+2         ;;STARTING ADDRESS OF TEST 33
12056  061162  030574                        .WORD   TST34+2         ;;STARTING ADDRESS OF TEST 34
12057  061164  031752                        .WORD   TST35+2         ;;STARTING ADDRESS OF TEST 35
12058                                 .SBTTL  ERROR HANDLER ROUTINE
12059
12060                                 ;;**************************************************************
12061                                 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
12062                                 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
12063                                 ;*AND GO TO ERRTYP ON ERROR
12064                                 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
12065                                 ;*SW15=1         HALT ON ERROR
12066                                 ;*SW13=1         INHIBIT ERROR TYPEOUTS
12067                                 ;*SW10=1         BELL ON ERROR
12068                                 ;*SW09=1         LOOP ON ERROR
12069                                 ;*CALL
12070                                 ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
12071
12072  061166                         $ERROR:
12073  061166  104410                        CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
12074  061170  105237  001117         7$:    INCB    $ERFLG          ;;SET THE ERROR FLAG
12075  061174  001775                        BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
12076  061176  013777  001116  117752        MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
12077  061204  032777  002000  117742        BIT     #BIT10,@SWR     ;;BELL ON ERROR?
12078  061212  001402                        BEQ     1$              ;;NO - SKIP
12079  061214  104401  001212                TYPE    ,$BELL          ;;RING BELL
12080  061220  005237  001126         1$:    INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
12081  061224  011637  001132                MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
12082  061230  162737  000002  001132        SUB     #2,$ERRPC
12083  061236  117737  117670  001130        MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
12084  061244  032777  020000  117702        BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
12085  061252  001004                        BNE     20$             ;;SKIP TYPEOUTS
12086  061254  004737  033176                JSR     PC,ERRTYP       ;;GO TO USER ERROR ROUTINE
12087  061260  104401  001217                TYPE    ,$CRLF
12088  061264                         20$:
12089  061264  122737  000001  001242        CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
12090  061272  001007                        BNE     2$              ;;NO,SKIP APT ERROR REPORT
12091  061274  113737  001130  061306        MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
12092  061302  004737  063230                JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
12093  061306     000         21$:    .BYTE   0
12094  061307     000                 .BYTE   0
12095  061310  000777         22$:    BR      22$             ;;APT ERROR LOOP
12096  061312  005777  117636         2$:    TST     @SWR            ;;HALT ON ERROR
12097  061316  100002                        BPL     3$              ;;SKIP IF CONTINUE
12098  061320  000000                        HALT                    ;;HALT ON ERROR!
12099  061322  104410                        CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
12100  061324  032777  001000  117622  3$:    BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
12101  061332  001402                        BEQ     4$              ;;BR IF NO
12102  061334  013716  001124                MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
12103  061340  005737  001210         4$:    TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
12104  061344  001402                        BEQ     5$              ;;BR IF NONE
12105  061346  013716  001210                MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
12106  061352                         5$:
12107  061352  022737  033156  000042        CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
12108  061360  001001                        BNE     6$              ;;BRANCH IF NO
```

```
12109  061362  000000                          HALT                    ;;YES
12110  061364                          6$:
12111  061364  000002                          RTI                     ;;RETURN
12112                                  .SBTTL  TTY INPUT ROUTINE
12113
12114                                  ;;**********************************************************************
12115                                  .ENABL  LSB
12116  061366  000000                 $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
12117  061370  000000                 $TKQIN: .WORD   0               ;;INPUT POINTER
12118  061372  000000                 $TKQOUT: .WORD  0               ;;OUTPUT POINTER
12119  061374  000001                 $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
12120           061375                 $TKQEND=.
12121           061376                 .EVEN
12122
12123                                  ;*TK INITIALIZE ROUTINE
12124                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
12125                                  ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
12126                                  ;
12127                                  ;*CALL:
12128                                  ;*      JSR     PC,$TKINT
12129                                  ;*      RETURN
12130                                  ;
12131  061376  005037  061366         $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
12132  061402  012737  061374  061370         MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
12133  061410  013737  061370  061372         MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
12134  061416  012737  061446  000060         MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
12135  061424  012737  000200  000062         MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
12136  061432  005777  117524                 TST     @$TKB           ;;CLEAR DONE FLAG
12137  061436  012777  000100  117514         MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
12138  061444  000207                         RTS     PC              ;;RETURN TO CALLER
12139
12140                                  ;*TK SERVICE ROUTINE
12141                                  ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
12142                                  ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
12143                                  ;*IT IN THE QUEUE.
12144                                  ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
12145                                  ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (SHUT2)
12146                                  ;
12147  061446  117746  117510         $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
12148  061452  042716  177600                 BIC     #^C177,(SP)     ;;STRIP THE JUNK
12149  061456  021627  000003                 CMP     (SP),#3         ;;IS IT A CONTROL C?
12150  061462  001007                         BNE     1$              ;;BRANCH IF NO
12151  061464  104401  062562                 TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
12152  061470  004737  061376                 JSR     PC,$TKINT       ;;INIT THE KEYBOARD
12153  061474  005726                         TST     (SP)+           ;;CLEAN UP STACK
12154  061476  000137  057364                 JMP     SHUT2           ;;CONTROL C RESTART
12155  061502  021627  000007         1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
12156  061506  001004                         BNE     2$              ;;BRANCH IF NO
12157  061510  022737  000176  001154         CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
12158  061516  001500                         BEQ     6$              ;;GO TO SWR CHANGE
12159
12160  061520                          2$:
12161  061520  022737  000001  061366         CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
12162  061526  001004                         BNE     3$              ;;BRANCH IF NO
12163  061530  104401  001212                 TYPE    ,$BELL          ;;RING THE TTY BELL
12164  061534  005726                         TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
```

```
12165  061536  000451                        BR      5$                ;;EXIT
12166  061540  021627  000023          3$:   CMP     (SP),#23          ;;IS IT A CONTROL-S?
12167  061544  001021                        BNE     32$               ;;BRANCH IF NO
12168  061546  005077  117406                CLR     @$TKS             ;;DISABLE TTY KEYBOARD INTERRUPTS
12169  061552  005726                        TST     (SP)+             ;;CLEAN CHAR OFF STACK
12170  061554  105777  117400          31$:  TSTB    @$TKS             ;;WAIT FOR A CHAR
12171  061560  100375                        BPL     31$               ;;LOOP UNTIL ITS THERE
12172  061562  117746  117374                MOVB    @$TKB,-(SP)       ;;GET THE CHARACTER
12173  061566  042716  177600                BIC     #^C177,(SP)       ;;MAKE IT 7-BIT ASCII
12174  061572  022627  000021                CMP     (SP)+,#21         ;;IS IT A CONTROL-Q?
12175  061576  001366                        BNE     31$               ;;BRANCH IF NO
12176  061600  012777  000100  117352        MOV     #100,@$TKS        ;;REENABLE TTY KEYBOARD INTERRUPTS
12177  061606  000002                        RTI                       ;;RETURN
12178  061610  005237  061366          32$:  INC     $TKCNT            ;;COUNT THIS CHARACTER
12179  061614  021627  000140                CMP     (SP),#140         ;;IS IT UPPER CASE?
12180  061620  002405                        BLT     4$                ;;BRANCH IF YES
12181  061622  021627  000175                CMP     (SP),#175         ;;IS IT A SPECIAL CHAR?
12182  061626  003002                        BGT     4$                ;;BRANCH IF YES
12183  061630  042716  000040                BIC     #40,(SP)          ;;MAKE IT UPPER CASE
12184  061634  112677  177530          4$:   MOVB    (SP)+,@$TKQIN     ;;AND PUT IT IN QUEUE
12185  061640  005237  061370                INC     $TKQIN            ;;UPDATE THE POINTER
12186  061644  023727  061370  061375        CMP     $TKQIN,#$TKQEND   ;;GO OFF THE END?
12187  061652  001003                        BNE     5$                ;;BRANCH IF NO
12188  061654  012737  061374  061370        MOV     #$TKQSRT,$TKQIN   ;;RESET THE POINTER
12189  061662  000002                  5$:   RTI                       ;;RETURN
12190
12191                                  ;;**********************************************************************
12192                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
12193                                  ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
12194                                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
12195                                  ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
12196  061664  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR        ;;IS THE SOFT-SWR SELECTED
12197  061672  001124                        BNE     15$               ;;EXIT IF NOT
12198  061674  105777  117260                TSTB    @$TKS             ;;IS A CHAR WAITING?
12199  061700  100121                        BPL     15$               ;;IF NOT, EXIT
12200  061702  117746  117254                MOVB    @$TKB,-(SP)       ;;YES
12201  061706  042716  177600                BIC     #^C177,(SP)       ;;MAKE IT  7-BIT ASCII
12202  061712  021627  000007                CMP     (SP),#7           ;;IS IT A CONTROL-G?
12203  061716  001300                        BNE     2$                ;;IF NOT, PUT IT IN THE TTY QUEUE
12204                                                                   ;;AND EXIT
12205
12206                                  ;;**********************************************************************
12207                                  ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
12208                                  ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
12209                                  ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
12210  061720  123727  001150  000001  6$:   CMPB    $AUTOB,#1         ;;ARE WE RUNNING IN AUTO-MODE?
12211  061726  001674                        BEQ     2$                ;;BRANCH IF YES
12212  061730  005726                        TST     (SP)+             ;;CLEAR CONTROL-G OFF STACK
12213  061732  004737  061376                JSR     PC,$TKINT         ;;FLUSH THE TTY INPUT QUEUE
12214  061736  005077  117216                CLR     @$TKS             ;;DISABLE TTY KEYBOARD INTERRUPTS
12215  061742  112737  000001  001151        MOVB    #1,$INTAG         ;;SET INTERRUPT MODE INDICATOR
12216
12217  061750  104401  062574                TYPE    ,$CNTLG           ;;ECHO THE CONTROL-G (^G)
12218  061754  104401  062601          $GTSWR: TYPE    ,$MSWR            ;;TYPE CURRENT CONTENTS
12219  061760  013746  000176                MOV     SWREG,-(SP)       ;;SAVE SWREG FOR TYPEOUT
12220  061764  104402                        TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
12221  061766  104401  062612                 TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
12222  061772  005046                 19$:    CLR     -(SP)           ;;CLEAR COUNTER
12223  061774  005046                         CLR     -(SP)           ;;THE NEW SWR
12224  061776  105777  117156          7$:    TSTB    @$TKS           ;;CHAR THERE?
12225  062002  100375                         BPL     7$              ;;IF NOT TRY AGAIN
12226
12227  062004  117746  117152                 MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
12228  062010  042716  177600                 BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
12229
12230  062014  021627  000003                 CMP     (SP),#3         ;;IS IT A CONTROL-C?
12231  062020  001015                         BNE     9$              ;;BRANCH IF NOT
12232  062022  104401  062562                 TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
12233  062026  062706  000006                 ADD     #6,SP           ;;CLEAN UP STACK
12234  062032  123727  001151  000001         CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
12235  062040  001003                         BNE     8$              ;;BRANCH IF NO
12236  062042  012777  000100  117110         MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
12237  062050  000137  057364          8$:    JMP     SHUT2           ;;CONTROL-C RESTART
12238
12239
12240  062054  021627  000025          9$:    CMP     (SP),#25        ;;IS IT A CONTROL-U?
12241  062060  001005                         BNE     10$             ;;BRANCH IF NOT
12242  062062  104401  062567                 TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
12243  062066  062706  000006          20$:   ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
12244  062072  000737                         BR      19$             ;;LET'S TRY IT AGAIN
12245
12246
12247  062074  021627  000015          10$:   CMP     (SP),#15        ;;IS IT A <CR>?
12248  062100  001022                         BNE     16$             ;;BRANCH IF NO
12249  062102  005766  000004                 TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
12250  062106  001403                         BEQ     11$             ;;BRANCH IF YES
12251  062110  016677  000002  117036         MOV     2(SP),@SWR      ;;SAVE NEW SWR
12252  062116  062706  000006          11$:   ADD     #6,SP           ;;CLEAR UP STACK
12253  062122  104401  001217          14$:   TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
12254  062126  123727  001151  000001         CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
12255  062134  001003                         BNE     15$             ;;BRANCH IF NOT
12256  062136  012777  000100  117014         MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
12257  062144  000002          15$:   RTI                             ;;RETURN
12258  062146  004737  060506          16$:   JSR     PC,$TYPEC       ;;ECHO CHAR
12259  062152  021627  000060                 CMP     (SP),#60        ;;CHAR < 0?
12260  062156  002420                         BLT     18$             ;;BRANCH IF YES
12261  062160  021627  000067                 CMP     (SP),#67        ;;CHAR > 7?
12262  062164  003015                         BGT     18$             ;;BRANCH IF YES
12263  062166  042726  000060                 BIC     #60,(SP)+       ;;STRIP-OFF ASCII
12264  062172  005766  000002                 TST     2(SP)           ;;IS THIS THE FIRST CHAR
12265  062176  001403                         BEQ     17$             ;;BRANCH IF YES
12266  062200  006316                         ASL     (SP)            ;;NO, SHIFT PRESENT
12267  062202  006316                         ASL     (SP)            ;;    CHAR OVER TO MAKE
12268  062204  006316                         ASL     (SP)            ;;    ROOM FOR NEW ONE.
12269  062206  005266  000002          17$:   INC     2(SP)           ;;KEEP COUNT OF CHAR
12270  062212  056616  177776                 BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
12271  062216  000667                         BR      7$              ;;GET THE NEXT ONE
12272  062220  104401  001216          18$:   TYPE    ,$QUES          ;;TYPE ?<CR><LF>
12273  062224  000720                         BR      20$             ;;SIMULATE CONTROL-U
12274                          .DSABL  LSB
12275
12276
```

```
12277                                   ;;************************************************************************
12278                                   ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
12279                                   ;*CALL:
12280                                   ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
12281                                   ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
12282                                   ;*                              ;;WITH PARITY BIT STRIPPED OFF
12283                                   ;
12284
12285  062226  011646                   $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
12286  062230  016666  000004  000002           MOV     4(SP),2(SP)     ;;THE PS
12287  062236  005066  000004                   CLR     4(SP)           ;;GET READY FOR A CHARACTER
12288  062242  005046                           CLR     -(SP)           ;;PUT NEW PS ON STACK
12289  062244  012746  062252                   MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
12290  062250  000002                           RTI                     ;;POP NEW PC AND PS
12291  062252                           64$:
12292  062252  005737  061366           1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
12293  062256  001775                           BEQ     1$
12294  062260  005337  061366                   DEC     $TKCNT          ;;DECREMENT THE COUNTER
12295  062264  117766  177102  000004           MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
12296  062272  005237  061372                   INC     $TKQOUT         ;;UPDATE THE POINTER
12297  062276  023727  061372  061375           CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
12298  062304  001003                           BNE     2$              ;;BRANCH IF NO
12299  062306  012737  061374  061372           MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
12300  062314  000002           2$:     RTI                             ;;RETURN
12301                                   ;;************************************************************************
12302                                   ;*THIS ROUTINE WILL !NPUT A STRING FROM THE TTY
12303                                   ;*CALL:
12304                                   ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
12305                                   ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
12306                                   ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
12307
12308  062316  010346                   $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
12309  062320  005046                           CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
12310  062322  012703  062552           1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
12311  062326  022703  062562           2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
12312  062332  101456                           BLOS    4$              ;;BR IF YES
12313  062334  104411                           RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
12314  062336  112613                           MOVB    (SP)+,(R3)      ;;GET CHARACTER
12315  062340  122713  000177           10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
12316  062344  001022                           BNE     5$              ;;BR IF NO
12317  062346  005716                           TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
12318  062350  001007                           BNE     6$              ;;BR IF NO
12319  062352  112737  000134  062550           MOVB    #'\,9$          ;;TYPE A BACK SLASH
12320  062360  104401  062550                   TYPE    ,9$
12321  062364  012716  177777                   MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
12322  062370  005303                   6$:     DEC     R3              ;;BACKUP BY ONE
12323  062372  020327  062552                   CMP     R3,#$TTYIN      ;;STACK EMPTY?
12324  062376  103434                           BLO     4$              ;;BR IF YES
12325  062400  111337  062550                   MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
12326  062404  104401  062550                   TYPE    ,9$             ;;GO TYPE
12327  062410  000746                           BR      2$              ;;GO READ ANOTHER CHAR.
12328  062412  005716                   5$:     TST     (SP)            ;;RUBOUT KEY SET?
12329  062414  001406                           BEQ     7$              ;;BR IF NO
12330  062416  112737  000134  062550           MOVB    #'\,9$          ;;TYPE A BACK SLASH
12331  062424  104401  062550                   TYPE    ,9$
12332  062430  005016                           CLR     (SP)            ;;CLEAR THE RUBOUT KEY
```

```
12333  062432  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
12334  062436  001003                          BNE     8$              ;;BR IF NO
12335  062440  104401  062567                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
12336  062444  000726                          BR      1$              ;;GO START OVER
12337  062446  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "^R"?
12338  062452  001011                          BNE     3$              ;;BRANCH IF NO
12339  062454  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
12340  062456  104401  001217                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
12341  062462  104401  062552                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
12342  062466  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
12343  062470  104401  001216          4$:     TYPE    ,$QUES          ;;TYPE A '?'
12344  062474  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
12345  062476  111337  062550          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
12346  062502  104401  062550                  TYPE    ,9$
12347  062506  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
12348  062512  001305                          BNE     2$              ;;LOOP IF NOT RETURN
12349  062514  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
12350  062520  104401  001220                  TYPE    ,$LF            ;;TYPE A LINE FEED
12351  062524  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
12352  062526  012603                          MOV     (SP)+,R3        ;;RESTORE R3
12353  062530  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
12354  062532  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
12355  062540  012766  062552  000004          MOV     #$TTYIN,4(SP)
12356  062546  000002                          RTI                     ;;RETURN
12357  062550     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
12358  062551     000                  .BYTE   0                       ;;TERMINATOR
12359  062552  000010          $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
12360  062562  041536  005015     000  $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL "C"
12361  062567     136  006525  000012  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
12362  062574  043536  005015     000  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
12363  062601     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
12364  062606  036440  000040
12365  062612  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
12366  062620  020075     000
12367          062624                  .EVEN
12368                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
12369
12370                                  ;;**********************************************************************
12371                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
12372                                  ;*CHANGE IT TO BINARY.
12373                                  ;*CALL:
12374                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
12375                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
12376                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
12377
12378  062624  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
12379  062626  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
12380  062634  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12381  062636  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12382  062640  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12383  062642  104412                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
12384  062644  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
12385  062646  005001                          CLR     R1              ;;CLEAR DATA WORD
12386  062650  005002                          CLR     R2
12387  062652  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
12388  062654  001412                          BEQ     3$              ;;IF ZERO GET OUT
```

```
12389  062656  006301                        ASL     R1              ;;*2
12390  062660  006102                        ROL     R2
12391  062662  006301                        ASL     R1              ;;*4
12392  062664  006102                        ROL     R2
12393  062666  006301                        ASL     R1              ;;*8
12394  062670  006102                        ROL     R2
12395  062672  042716  177770                BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
12396  062676  062601                        ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
12397  062700  000764                        BR      2$              ;;LOOP
12398  062702  005726                3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
12399  062704  010166  000012                MOV     R1,12(SP)       ;;SAVE THE RESULT
12400  062710  010237  062724                MOV     R2,$HIOCT
12401  062714  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
12402  062716  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
12403  062720  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
12404  062722  000002                        RTI                     ;;RETURN
12405  062724  000000                $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
12406                                 .SBTTL  TRAP DECODER
12407
12408                                 ;;**************************************************************
12409                                 ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
12410                                 ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
12411                                 ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
12412                                 ;*GO TO THAT ROUTINE.
12413
12414  062726  016646  000002        $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
12415  062732  042716  000020                BIC     #20,(SP)        ;; THE CALLER--DO NOT ALLOW
12416  062736  012746  062744                MOV     #1$,-(SP)       ;; T-BIT TRAPS
12417  062742  000002                        RTI                     ;;SET THE NEW STATUS
12418  062744  010046                1$:     MOV     R0,-(SP)        ;;SAVE R0
12419  062746  016600  000002                MOV     2(SP),R0        ;;GET TRAP ADDRESS
12420  062752  005740                        TST     -(R0)           ;;BACKUP BY 2
12421  062754  1110STRUCTION.
12437
12438                                 ;       ROUTINE
12439                                 ;       -------
12440  063000  062766                $TRPAD: .WORD   $TRAP2
12441  063002  060274                        $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
12442  063004  060072                        $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
12443  063006  060046                        $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
12444  063010  060106                        $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
```

```
12445  063012  057622                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
12446  063014  057546                          $TYPBN  ;;CALL=TYPBN    TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER
12447
12448  063016  061754                          $GTSWR  ;;CALL=GTSWR    TRAP+7(104407)  GET SOFT-SWR SETTING
12449
12450  063020  061664                          $CKSWR  ;;CALL=CKSWR    TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
12451  063022  062226                          $RDCHR  ;;CALL=RDCHR    TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
12452  063024  062316                          $RDLIN  ;;CALL=RDLIN    TRAP+12(104412) TTY TYPEIN STRING ROUTINE
12453  063026  062624                          $RDOCT  ;;CALL=RDOCT    TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
12454  063030  057452                          $SAVREG ;;CALL=SAVREG   TRAP+14(104414) SAVE R0-R5 ROUTINE
12455  063032  057510                          $RESREG ;;CALL=RESREG   TRAP+15(104415) RESTORE R0-R5 ROUTINE
12456                                  .SBTTL  POWER DOWN AND UP ROUTINES
12457
12458                                  ;;**********************************************************************
12459                                  ;POWER DOWN ROUTINE
12460  063034  012737  063174  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
12461  063042  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
12462  063050  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12463  063052  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12464  063054  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12465  063056  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
12466  063060  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
12467  063062  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
12468  063064  017746  116064                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
12469  063070  010637  063200                  MOV     SP,$SAVR6       ;;SAVE SP
12470  063074  012737  063106  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
12471  063102  000000                          HALT
12472  063104  000776                          BR      .-2             ;;HANG UP
12473
12474                                  ;;**********************************************************************
12475                                  ;POWER UP ROUTINE
12476  063106  012737  063174  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
12477  063114  013706  063200                  MOV     $SAVR6,SP       ;;GET SP
12478  063120  005037  063200                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
12479  063124  005237  063200          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
12480  063130  001375                          BNE     1$              ;;OF  WORD
12481  063132  012677  116016                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
12482  063136  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
12483  063140  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
12484  063142  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
12485  063144  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
12486  063146  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
12487  063150  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
12488  063152  012737  063034  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
12489  063160  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
12490  063166  104401                          TYPE                    ;REPORT THE POWER FAILURE
12491  063170  063202                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
12492  063172  000002                          RTI
12493  063174  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
12494  063176  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
12495  063200  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
12496  063202  005015  047520  042527  $POWER: .ASCIZ  <15><12>''POWER''
12497  063210  000122
12498                                          .EVEN
12499                                  .SBTTL  APT COMMUNICATIONS ROUTINE
12500
```

```
12501                                   ;;*****************************************************************
12502  063212  112737  000001  063456  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
12503  063220  112737  000001  063454  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
12504  063226  000403                  BR      $ATYC
12505  063230  112737  000001  063456  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
12506  063236                          $ATYC:
12507  063236  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12508  063240  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12509  063242  105737  063454          TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
12510  063246  001450                  BEQ     5$              ;;IF NOT:  BR
12511  063250  122737  000001  001242  CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
12512  063256  001031                  BNE     3$              ;;IF NOT:  BR
12513  063260  132737  000100  001243  BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
12514  063266  001425                  BEQ     3$              ;;IF NOT:  BR
12515  063270  017600  000004          MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
12516  063274  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
12517  063302  005737  001222          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
12518  063306  001375                  BNE     1$              ;;IF NOT:  WAIT
12519  063310  010037  001236          MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
12520  063314  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
12521  063316  001376                  BNE     2$
12522  063320  163700  001236          SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
12523  063324  006200                  ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
12524  063326  010037  001240          MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
12525  063332  012737  000004  001222  MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
12526  063340  000413                  BR      5$
12527  063342  017637  000004  063366  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
12528  063350  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
12529  063356  013746  177776          MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
12530  063362  004737  060274          JSR     PC,$TYPE        ;;CALL TYPE MACRO
12531  063366  000000                  4$:     .WORD   0
12532  063370                          5$:
12533  063370  105737  063456          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
12534  063374  001416                  BEQ     12$             ;;IF NOT:  BR
12535  063376  005737  001242          TST     $ENV            ;;RUNNING UNDER APT?
12536  063402  001413                  BEQ     12$             ;;IF NOT:  BR
12537  063404  005737  001222          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
12538  063410  001375                  BNE     11$             ;;IF NOT:  WAIT
12539  063412  017637  000004  001224  MOV     @4(SP),$FATAL   ;;GET ERROR #
12540  063420  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
12541  063426  005237  001222          INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
12542  063432  105037  063456          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
12543  063436  105037  063455          CLRB    $LFLG           ;;CLEAR LOG FLAG
12544  063442  105037  063454          CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
12545  063446  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
12546  063450  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
12547  063452  000207                  RTS     PC              ;;RETURN
12548  063454     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
12549  063455     000          $LFLG:  .BYTE   0               ;;LOG FLAG
12550  063456     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
12551          063460                  .EVEN
12552          000200          APTSIZE=200
12553          000001          APTENV=001
12554          000100          APTSPOOL=100
12555          000040          APTCSUP=040
12556
```

```
12557                                   .NLIST  BEX
```

```
                                .SBTTL  CONSOLE MESSAGES

063460                          SCTMSG:
063460  005015  040503  047116  .ASCII  <CR><LF>@CANNOT RECOVER THE BAD SECTOR FILES FROM LAST @<CR><LF>
063542  051124  041501  020113  .ASCIZ  @TRACK FOR THIS DEVICE@
063570     051                  CLSPRN: .ASCII  @)@
063571     075     000          EQUALS: .ASCIZ  @=@
063573     015  025012     000  PROMPT: .ASCIZ  <CR><LF>@*@
063577     077     000          QSTMRK: .ASCIZ  @?@
063601                          HELPQST:
063601     015  006412  052012  .ASCII <CR><LF><CR><LF>/THIS PROGRAM SHOULD BE HALTED BY TYPE ^C./
063656  005015  044124  020105  .ASCII <CR><LF>/THE PROGRAM WILL BE HALTED AT END OF PASS/<CR><LF><CR>
063734  005015  054524  042520  .ASCIZ  <CR><LF>@TYPE HELP TEXT (Y OR N)??@
063770                          UBUSQST:
063770  005015  044103  047101  .ASCIZ  <CR><LF>@CHANGE RM03 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N)<CR> ??@
064067     015  052412  042523  CNSL00: .ASCIZ  <CR><LF>@USE SAME DEVICES (Y OR N) ??@
064126  005015  046522  031460  CNSL01: .ASCIZ  <CR><LF>@RM03 BUS ADDRESS (@
064153     015  042412  052116  CNSL02: .ASCII  <CR><LF>@ENTRY NOT IN I/O PAGE@
064202  005015  042101  051104          .ASCIZ  <CR><LF>@ADDRESS MUST BE >160000@
064234  005015  046522  031460  CNSL03: .ASCIZ  <CR><LF>@RM03 VECTOR ADDRESS (@
064264  005015  047105  051124  CNSL04: .ASCII  <CR><LF>@ENTRY OUT OF RANGE@
064310  005015  042101  051104          .ASCIZ  <CR><LF>@ADDRESS MUST BE <1000@
064340  005015  046522  031460  CNSL05: .ASCIZ  <CR><LF>@RM03 INTERRUPT PRIORITY (@
064374  005015  047105  051124  CNSL06: .ASCIZ  <CR><LF>@ENTRY OUT OF RANGE@
064421                          CNSL07:
064421     015  052012  050131  .ASCII  <CR><LF>@TYPE (A) TO TEST ALL DEVICES, OR TYPE DEVICE@
064477     040  052516  041115  .ASCII  @ NUMBER(S)@
064511     015  052012  051105  .ASCIZ  <CR><LF>@TERMINATE INPUT WITH CARRIAGE RETURN@
064560  005015  044103  047101  XDPMG:  .ASCII  <CR><LF>/CHANGE XXDP PACK,CLEAR LOC 40/
064617     015  051012  051505          .ASCIZ  <CR><LF>/RESTART THE PROGRAM/
064645     015  047012  052117  NOTEX:  .ASCIZ  <CR><LF>/NOT EXIST DRIVE  /

064672                          .EVEN
```

```
                        .SBTTL  FUNCTION CODE TABLE

                        ;THE FUNCTION CODE TABLE IS USED TO DEFINE STATUS CONDITIONS FOR
                        ;EACH FUNCTION CODE.  BIT USAGE IS AS FOLLOWS:

                        ;       ATA - BIT 15 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF ATA SHOULD BE SET WHEN THE FUNCTION CODE IS EXECUTED, OTHERWISE,
                        ;BIT 15 IS ZERO, INDICATING THAT ATA SHOULD NOT NORMALLY BE SET.
                        ;NOTE THAT ATA MAY BE SET WHEN A COMMAND IS EXECUTED EVEN THOUGH
                        ;IT IS NOT EXPECTED AS A RESULT OF THE COMMAND.

                        ;       WCE - BIT 14 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF WRITE CHECK ERRORS ARE ENABLED AS A FUNCTION OF THE COMMAND.

                        ;       OPI - BIT 13 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF OPI ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.

                        ;       IVC - BIT 12 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF IVC ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.

                        ;       WLE - BIT 11 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF WRITE ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.
                        ;THE WRITE ERRORS WHICH ARE ENABLED ARE "WLE", "WCF", "DPE", "UPE".

                        ;       IAE - BIT 10 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF INVALID ADDRESS ERROR IS ENABLED FOR THAT COMMAND.

                        ;       AOE - BIT 09 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF READ AND WRITE ERRORS ARE ENABLED DURING THE EXECUTION OF THE
                        ;COMMAND.  THE ERRORS ENABLED BY THIS BIT ARE "TRE", "DLT", "NEM",
                        ;"MXF", "LBT", AND "AOE".

                        ;               BIT 08 IS NOT USED.

                        ;       HCE - BIT 07 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF HEADER ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.
                        ;HEADER ERRORS INCLUDE "HCRC", "HCE", "FER", AND "BSE".

                        ;       ECH - BIT 06 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF DATA FIELD ERRORS ARE ENABLED DURING THE EXECUTION OF THAT
                        ;COMMAND.  THESE ERRORS INCLUDE "MDPE", "DCK", AND "ECH".

                        ;               BIT 05 IS NOT USED.

                        ;               BIT 04 IS NOT USED.

                        ;               BIT 03 IS NOT USED.

                        ;               BIT 02 IS NOT USED.

                        ;               BIT 01 IS NOT USED.

                        ;       ILF - BIT 00 IS SET IF THE FUNCTION CODE IS ILLEGAL.

064672                  FNCDTB:                                 ;FUNCTION CODE TABLE
```

```
064672  020000          .WORD   OPI                                     ;NOP
064674  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (2)
064676  132000          .WORD   ATA!OPI!IVC!IAE                         ;SEEK
064700  130000          .WORD   ATA!OPI!IVC                             ;RECALIBRATE
064702  020000          .WORD   OPI                                     ;DRIVE CLEAR
064704  030000          .WORD   OPI!IVC                                 ;RELEASE
064706  130000          .WORD   OPI!ATA!IVC                             ;OFFSET
064710  130000          .WORD   OPI!ATA!IVC                             ;RETURN TO CENTERLINE
064712  020000          .WORD   OPI                                     ;READ IN PRESET
064714  020000          .WORD   OPI                                     ;PACK ACKNOWLEDGE
064716  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (24)
064720  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (26)
064722  132000          .WORD   ATA!OPI!IVC!IAE                         ;SEARCH
064724  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (32)
064726  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (34)
064730  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (36)
064732  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (40)
064734  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (42)
064736  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (44)
064740  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (46)
064742  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK DATA
064744  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK HEADER AND DATA
064746  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (54)
064750  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (56)
064752  037200          .WORD   OPI!IVC!WLE!IAE!AOE!HCE                 ;WRITE DATA
064754  037000          .WORD   OPI!IVC!WLE!IAE!AOE                     ;WRITE HEADER AND DATA
064756  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (64)
064760  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (66)
064762  033300          .WORD   OPI!IVC!IAE!AOE!HCF!ECH                 ;READ DATA
064764  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH                 ;READ HEADER AND DATA
064766  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (74)
064770  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (76)
```

```
                        .SBTTL  ATTENTION (ATA) TABLE

064772  001             ATNTBL: .BYTE   1.
064773  002                     .BYTE   2.
064774  004                     .BYTE   4.
064775  010                     .BYTE   8.
064776  020                     .BYTE   16.
064777  040                     .BYTE   32.
065000  100                     .BYTE   64.
065001  200                     .BYTE   128.
```

```
                                .SBTTL  DATA PATTERN TABLE

065002                          RGDTPT:
065002                          MIXED:
065002  000000                          .WORD   0.
065004  000001                          .WORD   1.
065006  000003                          .WORD   3.
065010  000007                          .WORD   7.
065012  000017                          .WORD   15.
065014  000037                          .WORD   31.
065016  000077                          .WORD   63.
065020  000177                          .WORD   127.
065022  000377                          .WORD   255.
065024  000777                          .WORD   511.
065026  001777                          .WORD   1023.
065030  003777                          .WORD   2047.
065032  007777                          .WORD   4095.
065034  017777                          .WORD   8191.
065036  037777                          .WORD   16383.
065040  077777                          .WORD   32767.
065042  177777                  ONES:   .WORD   65535.
065044  177777                          .WORD   65535.
065046  077777                          .WORD   32767.
065050  037777                          .WORD   16383.
065052  017777                          .WORD   8191.
065054  007777                          .WORD   4095.
065056  003777                          .WORD   2047.
065060  001777                          .WORD   1023.
065062  000777                          .WORD   511.
065064  000377                          .WORD   255.
065066  000177                          .WORD   127.
065070  000077                          .WORD   63.
065072  000037                          .WORD   31.
065074  000017                          .WORD   15.
065076  000007                          .WORD   7.
065100  000003                          .WORD   3.
065102  000001                          .WORD   1.
065104  000000                  ZEROS:  .WORD   0.
065106  000000                          .WORD   0.
065110  000001                          .WORD   1.
065112  000002                          .WORD   2.
065114  000004                          .WORD   4.
065116  000010                          .WORD   8.
065120  000020                          .WORD   16.
065122  000040                          .WORD   32.
065124  000100                          .WORD   64.
065126  000200                          .WORD   128.
065130  000400                          .WORD   256.
065132  001000                          .WORD   512.
065134  002000                          .WORD   1024.
065136  004000                          .WORD   2048.
065140  010000                          .WORD   4096.
065142  020000                          .WORD   8192.
065144  040000                          .WORD   16384.
065146  100000                          .WORD   32768.
065150  100000                          .WORD   32768.
```

```
065152  040000                          .WORD   16384.
065154  020000                          .WORD   8192.
065156  010000                          .WORD   4096.
065160  004000                          .WORD   2048.
065162  002000                          .WORD   1024.
065164  001000                          .WORD   512.
065166  000400                          .WORD   256.
065170  000200                          .WORD   128.
065172  000100                          .WORD   64.
065174  000040                          .WORD   32.
065176  000020                          .WORD   16.
065200  000010                          .WORD   8.
065202  000004                          .WORD   4.
065204  000002                          .WORD   2.
065206  000001                          .WORD   1.
065210  000000                          .WORD   0.
065212  177777                          .WORD   65535.
065214  177776                          .WORD   65534.
065216  177774                          .WORD   65532.
065220  177770                          .WORD   65528.
065222  177760                          .WORD   65520.
065224  177740                          .WORD   65504.
065226  177700                          .WORD   65472.
065230  177600                          .WORD   65408.
065232  177400                          .WORD   65280.
065234  177000                          .WORD   65024.
065236  176000                          .WORD   64512.
065240  174000                          .WORD   63488.
065242  170000                          .WORD   61440.
065244  160000                          .WORD   57344.
065246  140000                          .WORD   49152.
065250  100000                          .WORD   32768.
065252  000000                          .WORD   0.
065254  000000                          .WORD   0.
065256  100000                          .WORD   32768.
065260  140000                          .WORD   49152.
065262  160000                          .WORD   57344.
065264  170000                          .WORD   61440.
065266  174000                          .WORD   63488.
065270  176000                          .WORD   64512.
065272  177000                          .WORD   65024.
065274  177400                          .WORD   65280.
065276  177600                          .WORD   65408.
065300  177700                          .WORD   65472.
065302  177740                          .WORD   65504.
065304  177760                          .WORD   65520.
065306  177770                          .WORD   65528.
065310  177774                          .WORD   65532.
065312  177776                          .WORD   65534.
065314  177777                          .WORD   65535.
065316  125252          EARLY:          .WORD   43690.
065320  152525                          .WORD   43690./2
065322  125252                          .WORD   43690.
065324  177777                          .WORD   65535.
065326  177776                          .WORD   65534.
065330  177775                          .WORD   65533.
```

```
065332  177773                  .WORD   65531.
065334  177767                  .WORD   65527.
065336  177757                  .WORD   65519.
065340  177737                  .WORD   65503.
065342  177677                  .WORD   65471.
065344  177577                  .WORD   65407.
065346  177377                  .WORD   65279.
065350  176777                  .WORD   65023.
065352  175777                  .WORD   64511.
065354  173777                  .WORD   63487.
065356  167777                  .WORD   61439.
065360  157777                  .WORD   57343.
065362  137777                  .WORD   49151.
065364  077777                  .WORD   32767.
065366  077777                  .WORD   32767.
065370  137777                  .WORD   49151.
065372  157777                  .WORD   57343.
065374  167777                  .WORD   61439.
065376  173777                  .WORD   63487.
065400  175777                  .WORD   64511.
065402  176777                  .WORD   65023.
065404  177377                  .WORD   65279.
065406  177577                  .WORD   65407.
065410  177677                  .WORD   65471.
065412  177737                  .WORD   65503.
065414  177757                  .WORD   65519.
065416  177767                  .WORD   65527.
065420  177773                  .WORD   65531.
065422  177775                  .WORD   65533.
065424  177776                  .WORD   65534.
065426  177777                  .WORD   65535.
065430                  ENRGDT:
```

```
                                .SBTTL  ERROR MESSAGE TABLE

065430  072024  000000          EMT1:   .WORD   EMS1,0
065434  072073  072110  000000  EMT2:   .WORD   EMS2,EMS3,0
065442  072073  072153  000000  EMT3:   .WORD   EMS2,EMS4,0
065450  072216  072246  000000  EMT4:   .WORD   EMS5,EMS6,0
065456  072216  072360  000000  EMT5:   .WORD   EMS5,EMS10,0
065464  077053  074241  000000  EMT6:   .WORD   EMS167,EMS64,0
065472  075007  077100  000000  EMT7:   .WORD   EMS110,EMS170,0
065500  072313  000000          EMT10:  .WORD   EMS7,0
065504  072360  000000          EMT11:  .WORD   EMS10,0
065510  072422  072433  000000  EMT12:  .WORD   EMS11,EMS12,0
065516  072474  072505  072516  EMT13:  .WORD   EMS13,EMS14,EMS15,EMS16,0
065530  072570  074241  000000  EMT14:  .WORD   EMS17,EMS64,0
065536  072422  072653  000000  EMT15:  .WORD   EMS11,EMS21,0
065544  072422  072676  073025  EMT16:  .WORD   EMS11,EMS22,EMS27,0
065554  072422  072712  073036  EMT17:  .WORD   EMS11,EMS23,EMS30,0
065564  072422  072740  073036  EMT20:  .WORD   EMS11,EMS24,EMS30,0
065574  072422  072767  073025  EMT21:  .WORD   EMS11,EMS25,EMS27,0
065604  072422  073004  073025  EMT22:  .WORD   EMS11,EMS26,EMS27,0
065614  072422  073046  073036  EMT23:  .WORD   EMS11,EMS31,EMS30,0
065624  072422  073075  073036  EMT24:  .WORD   EMS11,EMS32,EMS30,0
065634  072422  073124  073036  EMT25:  .WORD   EMS11,EMS33,EMS30,0
065644  072422  073152  073036  EMT26:  .WORD   EMS11,EMS34,EMS30,0
065654  072422  073223  073036  EMT27:  .WORD   EMS11,EMS35,EMS30,0
065664  072422  073252  073036  EMT30:  .WORD   EMS11,EMS36,EMS30,0
065674  072422  073301  073036  EMT31:  .WORD   EMS11,EMS37,EMS30,0
065704  072422  073327  073036  EMT32:  .WORD   EMS11,EMS40,EMS30,0
065714  072422  073356  073036  EMT33:  .WORD   EMS11,EMS41,EMS30,0
065724  072422  073404  073036  EMT34:  .WORD   EMS11,EMS42,EMS30,0
065734  072422  073433  073036  EMT35:  .WORD   EMS11,EMS43,EMS30,0
065744  072422  073462  073036  EMT36:  .WORD   EMS11,EMS44,EMS30,0
065754  072422  073535  073036  EMT37:  .WORD   EMS11,EMS45,EMS30,0
065764  074325  072630  000000  EMT40:  .WORD   EMS66,EMS20,0
065772  074513  076221  074521  EMT41:  .WORD   EMS75,EMS141,EMS76,0
066002  076312  076322  074447  EMT42:  .WORD   EMS144,EMS145,EMS72,EMS76,0
066014  073627  073745  074521  EMT43:  .WORD   EMS47,EMS53,EMS76,0
066024  074552  073745  074521  EMT44:  .WORD   EMS77,EMS53,EMS76,0
066034  074600  073745  074521  EMT45:  .WORD   EMS100,EMS53,EMS76,0
066044  074626  073745  074521  EMT46:  .WORD   EMS101,EMS53,EMS76,0
066054  074257  074241  000000  EMT47:  .WORD   EMS65,EMS64,0
066062  072474  072516  074213  EMT50:  .WORD   EMS13,EMS15,EMS63,0
066072  072422  073600  073036  EMT51:  .WORD   EMS11,EMS46,EMS30,EMS67,0
066104  073627  073745  074375  EMT52:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS140,EMS141,0
066122  073627  073745  074375  EMT53:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS141,EMS164,0
066140  073656  073745  074375  EMT54:  .WORD   EMS50,EMS53,EMS67,0
066150  076247  076264  073745  EMT55:  .WORD   EMS142,EMS143,EMS53,EMS67,0
066162  073705  074447  074375  EMT56:  .WORD   EMS51,EMS72,EMS67,EMS115,EMS50,EMS70,0
066200  077053  074457  074375  EMT57:  .WORD   EMS167,EMS73,EMS67,0
066210  074030  074375  075207  EMT60:  .WORD   EMS56,EMS67,EMS115,EMS150,EMS152,EMS70,0
066226  074434  074030  074375  EMT61:  .WORD   EMS71,EMS56,EMS67,EMS115,EMS150,EMS152,EMS72,0
066246  076200  074375  075207  EMT62:  .WORD   EMS140,EMS67,EMS115,EMS47,EMS70,0
066262  000000                  EMT63:  .WORD
066264  076405  076450  074422  EMT64:  .WORD   EMS150,EMS152,EMS70,EMS115,EMS56,EMS73,EMS67,0
066304  073733  077500  077661  EMT65:  .WORD   EMS52,EMS205,EMS214,EMS206,EMS115,EMS51,EMS72,0
066324  077012  074703  074447  EMT66:  .WORD   EMS165,EMS103,EMS72,EMS124,0
```

```
066336  077012  074703  074447  EMT67:  .WORD   EMS165,EMS103,EMS72,EMS171,0
066350  073600  072630  076705  EMT70:  .WORD   EMS46,EMS20,EMS163,0
066360  074434  074626  076705  EMT71:  .WORD   EMS71,EMS101,EMS163,0
066370  073627  074447  076705  EMT72:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS140,EMS141,0
066406  073627  074447  076705  EMT73:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS141,EMS72,0
066424  074030  073745  076705  EMT74:  .WORD   EMS56,EMS53,EMS163,0
066434  074434  074030  076705  EMT75:  .WORD   EMS71,EMS56,EMS163,EMS115,EMS150,EMS152,EMS72,0
066454  073656  073745  076705  EMT76:  .WORD   EMS50,EMS53,EMS163,0
066464  076247  076264  073745  EMT77:  .WORD   EMS142,EMS143,EMS53,EMS163,0
066476  074513  076221  076705  EMT100: .WORD   EMS75,EMS141,EMS163,EMS115,EMS47,EMS70,0
066514  074513  076405  076705  EMT101: .WORD   EMS75,EMS150,EMS163,EMS115,EMS56,EMS73,0
066532  077053  074457  076705  EMT102: .WORD   EMS167,EMS73,EMS163,0
066542  076370  076424  074474  EMT103: .WORD   EMS147,EMS151,EMS74,EMS163,0
066554  073705  074474  076705  EMT104: .WORD   EMS51,EMS74,EMS163,EMS115,EMS50,EMS70,0
066572  077012  074600  076705  EMT105: .WORD   EMS165,EMS100,EMS163,0
066602  077012  074552  076705  EMT106: .WORD   EMS165,EMS77,EMS163,0
066612  076312  076322  073745  EMT107: .WORD   EMS144,EMS145,EMS53,EMS163,EMS115,EMS143,EMS70,0
066632  075007  075047  075214  EMT110: .WORD   EMS110,EMS112,EMS116,EMS111,0
066644  075133  072153  000000  EMT111: .WORD   EMS113,EMS4,0
066652  075133  072110  000000  EMT112: .WORD   EMS113,EMS3,0
066660  075007  075207  075214  EMT113: .WORD   EMS110,EMS115,EMS116,EMS117,EMS114,0
066674  075133  075266  000000  EMT114: .WORD   EMS113,EMS120,0
066702  075303  075743  075767  EMT115: .WORD   EMS121,EMS132,EMS133,0
066712  075346  075743  075767  EMT116: .WORD   EMS122,EMS132,EMS133,0
066722  075403  075743  075767  EMT117: .WORD   EMS123,EMS132,EMS133,0
066732  075446  075743  075767  EMT120: .WORD   EMS124,EMS132,EMS133,
066742  075500  075743  075767  EMT121: .WORD   EMS125,EMS132,EMS133,0
066752  075543  075743  075767  EMT122: .WORD   EMS126,EMS132,EMS133,0
066762  076143  075743  075767  EMT123: .WORD   EMS137,EMS132,EMS133,0
066772  075645  075743  075767  EMT124: .WORD   EMS130,EMS132,EMS133,0
067002  075703  075743  075767  EMT125: .WORD   EMS131,EMS132,EMS133,0
067012  075303  075743  076012  EMT126: .WORD   EMS121,EMS132,EMS134,EMS123,0
067024  075346  075743  076012  EMT127: .WORD   EMS122,EMS132,EMS134,EMS123,0
067036  075403  075743  076012  EMT130: .WORD   EMS123,EMS132,EMS134,EMS123,0
067050  075446  075743  076012  EMT131: .WORD   EMS124,EMS132,EMS134,EMS123,0
067062  075500  075743  076012  EMT132: .WORD   EMS125,EMS132,EMS134,EMS123,0
067074  075543  075743  076012  EMT133: .WORD   EMS126,EMS132,EMS134,EMS123,0
067106  076143  075743  076012  EMT134: .WORD   EMS137,EMS132,EMS134,EMS123,0
067120  075645  075743  076012  EMT135: .WORD   EMS130,EMS132,EMS134,EMS123,0
067132  075703  075743  076012  EMT136: .WORD   EMS131,EMS132,EMS134,EMS123,0
067144  075303  075743  076054  EMT137: .WORD   EMS121,EMS132,EMS135,0
067154  075403  075743  076054  EMT140: .WORD   EMS123,EMS132,EMS135,0
067164  075303  075743  076116  EMT141: .WORD   EMS121,EMS132,EMS136,0
067174  076143  075743  076116  EMT142: .WORD   EMS137,EMS132,EMS136,0
067204  075446  075743  076116  EMT143: .WORD   EMS124,EMS132,EMS136,0
067214  075500  075743  076116  EMT144: .WORD   EMS125,EMS132,EMS136,0
067224  075543  075743  076116  EMT145: .WORD   EMS126,EMS132,EMS136,0
067234  075703  075743  076116  EMT146: .WORD   EMS131,EMS132,EMS136,0
067244  076652  075743  076116  EMT147: .WORD   EMS162,EMS132,EMS136,0
067254  075645  075743  076116  EMT150: .WORD   EMS130,EMS132,EMS136,0
067264  076200  075207  076221  EMT151: .WORD   EMS140,EMS115,EMS141,EMS70,0
067276  076247  075207  076264  EMT152: .WORD   EMS142,EMS115,EMS143,EMS72,0
067310  076312  076322  076351  EMT153: .WORD   EMS144,EMS145,EMS146,EMS115,EMS143,EMS72,0
067326  076312  076322  072630  EMT154: .WORD   EMS144,EMS145,EMS20,EMS115,EMS143,EMS70,0
067344  076405  076450  076516  EMT155: .WORD   EMS150,EMS152,EMS154,EMS155,0
067356  076370  076424  076516  EMT156: .WORD   EMS147,EMS151,EMS154,EMS155,0
```

```
067370  076370  076424  076552  EMT157: .WORD   EMS147,EMS151,EMS150,EMS157,0
067402  076622  076552  076605  EMT160: .WORD   EMS161,EMS156,EMS160,0
067412  072767  073025  076552  EMT161: .WORD   EMS25,EMS27,EMS156,EMS160,0
067424  073004  073025  076552  EMT162: .WORD   EMS26,EMS27,EMS156,EMS160,0
067436  073627  076705  076200  EMT163: .WORD   EMS47,EMS163,EMS140,0
067446  073627  072630  000000  EMT164: .WORD   EMS47,EMS20,0
067454  074030  072630  000000  EMT165: .WORD   EMS56,EMS20,0
067462  073600  072630  000000  EMT166: .WORD   EMS46,EMS20,0
067470  100201  072630  000000  EMT167: .WORD   EMS224,EMS20,0
067476  077053  076516  077026  EMT170: .WORD   EMS167,EMS154,EMS166,0
067506  077053  076516  076570  EMT171: .WORD   EMS167,EMS154,EMS157,0
067516  077053  076516  076532  EMT172: .WORD   EMS167,EMS154,EMS155,0
067526  077127  075743  075767  EMT173: .WORD   EMS171,EMS132,EMS133,0
067536  077127  075743  076012  EMT174: .WORD   EMS171,EMS132,EMS134,EMS123,0
067550  075133  077302  000000  EMT175: .WORD   EMS113,EMS177,0
067556  077324  077341  000000  EMT176: .WORD   EMS200,EMS201,0
067564  077437  076221  073036  EMT177: .WORD   EMS203,EMS141,EMS30,EMS202,0
067576  077437  073705  073036  EMT200: .WORD   EMS203,EMS51,EMS30,EMS202,0
067610  077437  077452  073036  EMT201: .WORD   EMS203,EMS204,EMS30,EMS202,0
067622  077437  073656  073036  EMT202: .WORD   EMS203,EMS50,EMS30,EMS202,0
067634  077437  076264  073036  EMT203: .WORD   EMS203,EMS143,EMS30,EMS202,0
067646  076405  074474  077407  EMT204: .WORD   EMS150,EMS74,EMS202,EMS115,EMS152,EMS72,0
067664  073656  074703  074447  EMT205: .WORD   EMS50,EMS103,EMS72,0
067674  074712  074457  077407  EMT206: .WORD   EMS104,EMS73,EMS202,0
067704  074513  076221  075207  EMT207: .WORD   EMS75,EMS141,EMS115,EMS140,0
067716  074513  076405  000000  EMT210: .WORD   EMS75,EMS150,0
067724  073705  074447  075207  EMT211: .WORD   EMS51,EMS72,EMS115,EMS50,EMS70,0
067740  076247  073745  075207  EMT212: .WORD   EMS142,EMS53,EMS115,EMS143,EMS72,0
067754  073656  074703  073745  EMT213: .WORD   EMS50,EMS103,EMS53,0
067764  073733  077500  077026  EMT214: .WORD   EMS52,EMS205,EMS166,EMS206,EMS115,EMS51,EMS72,0
070004  073733  075244  074030  EMT215: .WORD   EMS52,EMS117,EMS56,EMS163,0
070016  074030  073036  074241  EMT216: .WORD   EMS56,EMS30,EMS64,0
070026  073600  073036  074241  EMT217: .WORD   EMS46,EMS30,EMS64,0
070036  073046  073036  074241  EMT220: .WORD   EMS31,EMS30,EMS64,0
070046  073627  073036  074241  EMT221: .WORD   EMS47,EMS30,EMS64,0
070056  073733  075244  074552  EMT222: .WORD   EMS52,EMS117,EMS77,0
070066  073733  075244  074213  EMT223: .WORD   EMS52,EMS117,EMS63,0
070076  000000                  EMT224: .WORD
070100  000000                  EMT225: .WORD
070102  000000                  EMT226: .WORD
070104  000000                  EMT227: .WORD
070106  000000                  EMT230: .WORD
070110  000000                  EMT231: .WORD
070112  000000                  EMT232: .WORD
070114  000000                  EMT233: .WORD
070116  000000                  EMT234: .WORD
070120  000000                  EMT235: .WORD
070122  000000                  EMT236: .WORD
070124  000000                  EMT237: .WORD
070126  000000                  EMT240: .WORD
070130  000000                  EMT241: .WORD
070132  000000                  EMT242: .WORD
070134  000000                  EMT243: .WORD
070136  000000                  EMT244: .WORD
070140  000000                  EMT245: .WORD
070142  077053  075743  077537  EMT246: .WORD   EMS167,EMS132,EMS207,0
```

```
070152  077053  075743  077564  EMT247: .WORD   EMS167,EMS132,EMS210,EMS125,0
070164  077053  077575  077564  EMT250: .WORD   EMS167,EMS211,EMS210,EMS207,EMS206,0
070200  077613  077636  000000  EMT251: .WORD   EMS212,EMS213,0
070206  077012  077613  000000  EMT252: .WORD   EMS165,EMS212,0
070214  076370  076424  076552  EMT253: .WORD   EMS147,EMS151,EMS156,EMS210,EMS26,EMS27,0
070232  077437  074626  073036  EMT254: .WORD   EMS203,EMS101,EMS30,0
070242  077437  077053  073036  EMT255: .WORD   EMS203,EMS167,EMS30,0
070252  077437  074600  073036  EMT256: .WORD   EMS203,EMS100,EMS30,0
070262  072422  073600  073036  EMT257: .WORD   EMS11,EMS46,EMS30,EMS102,0
070274  073733  075244  074030  EMT260: .WORD   EMS52,EMS117,EMS56,EMS102,0
070306  073733  077500  100003  EMT261: .WORD   EMS52,EMS205,EMS220,EMS206,EMS115,EMS51,EMS72,0
070326  074712  074457  077407  EMT262: .WORD   EMS104,EMS73,EMS202,0
070336  073656  073745  074654  EMT263: .WORD   EMS50,EMS53,EMS102,0
070346  074030  073745  074654  EMT264: .WORD   EMS56,EMS53,EMS102,EMS115,EMS150,EMS152,EMS70,0
070366  074434  074030  074654  EMT265: .WORD   EMS71,EMS56,EMS102,EMS115,EMS150,EMS152,EMS72,0
070406  076247  076264  073745  EMT266: .WORD   EMS142,EMS143,EMS53,EMS102,0
070420  073656  076351  075207  EMT267: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,0
070436  073627  073745  074654  EMT270: .WORD   EMS47,EMS53,EMS102,EMS115,EMS140,0
070452  073627  073745  074654  EMT271: .WORD   EMS47,EMS53,EMS102,EMS115,EMS141,EMS72,0
070470  074513  076221  074654  EMT272: .WORD   EMS75,EMS141,EMS102,EMS115,EMS47,EMS73,0
070506  073705  074474  074654  EMT273: .WORD   EMS51,EMS74,EMS102,EMS115,EMS50,EMS70,0
070524  074213  073745  074105  EMT274: .WORD   EMS63,EMS53,EMS57,EMS115,EMS41,EMS146,0
070542  075214  073745  073356  EMT275: .WORD   EMS116,EMS53,EMS41,EMS57,0
070554  073627  073745  074105  EMT276: .WORD   EMS47,EMS53,EMS57,EMS115,EMS140,0
070570  073627  073745  074105  EMT277: .WORD   EMS47,EMS53,EMS57,EMS115,EMS141,EMS72,0
070606  074030  073745  074105  EMT300: .WORD   EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS70,0
070626  074434  074030  073745  EMT301: .WORD   EMS71,EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS72,0
070650  077012  074600  074703  EMT302: .WORD   EMS165,EMS100,EMS103,EMS57,0
070662  077012  074626  074703  EMT303: .WORD   EMS165,EMS101,EMS103,EMS57,0
070674  077012  074552  074703  EMT304: .WORD   EMS165,EMS77,EMS103,EMS57,0
070706  073600  073036  074241  EMT305: .WORD   EMS46,EMS30,EMS64,EMS57,0
070720  073656  073745  074105  EMT306: .WORD   EMS50,EMS53,EMS57,0
070730  073656  076351  075207  EMT307: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,EMS57,0
070750  076247  076264  073745  EMT310: .WORD   EMS142,EMS143,EMS53,EMS57,0
070762  074712  074703  073745  EMT311: .WORD   EMS104,EMS103,EMS53,EMS57,0
070774  074741  074703  073745  EMT312: .WORD   EMS105,EMS103,EMS53,EMS57,0
071006  076312  076322  074703  EMT313: .WORD   EMS144,EMS145,EMS103,EMS57,EMS115,EMS143,EMS70,0
071026  073404  074703  073745  EMT314: .WORD   EMS42,EMS103,EMS53,EMS57,0
071040  073046  074703  073745  EMT315: .WORD   EMS31,EMS103,EMS53,EMS57,0
071052  074434  073046  074703  EMT316: .WORD   EMS71,EMS31,EMS103,EMS57,0
071064  073433  074703  074105  EMT317: .WORD   EMS43,EMS103,EMS57,0
071074  073535  074703  074105  EMT320: .WORD   EMS45,EMS103,EMS57,0
071104  073462  074703  074105  EMT321: .WORD   EMS44,EMS103,EMS57,0
071114  074770  072630  000000  EMT322: .WORD   EMS106,EMS20,0
071122  073252  074703  074105  EMT323: .WORD   EMS36,EMS103,EMS57,0
071132  077202  073252  074703  EMT324: .WORD   EMS173,EMS36,EMS103,EMS57,0
071144  077162  073252  074703  EMT325: .WORD   EMS172,EMS36,EMS103,EMS57,0
071156  072474  077217  072516  EMT326: .WORD   EMS13,EMS174,EMS15,EMS35,EMS53,EMS175,0
071174  076370  076424  074474  EMT327: .WORD   EMS147,EMS151,EMS74,EMS175,0
071206  074325  073745  077225  EMT330: .WORD   EMS66,EMS53,EMS175,0
071216  073124  074703  073745  EMT331: .WORD   EMS33,EMS103,EMS53,EMS175,0
071230  073327  074703  073745  EMT332: .WORD   EMS40,EMS103,EMS53,EMS57,0
071242  073705  074474  074105  EMT333: .WORD   EMS51,EMS74,EMS57,EMS115,EMS50,EMS70,0
071260  074513  076221  074105  EMT334: .WORD   EMS75,EMS141,EMS57,EMS115,EMS47,EMS73,0
071276  074513  076405  076450  EMT335: .WORD   EMS75,EMS150,EMS152,EMS57,EMS115,EMS56,EMS73,0
071316  074133  074146  074175  EMT336: .WORD   EMS60,EMS61,EMS62,0
```

```
071326  075214  075244  073152  EMT337: .WORD   EMS116,EMS117,EMS34,0
071336  073152  073745  073757  EMT340: .WORD   EMS34,EMS53,EMS54,EMS111,0
071350  074001  073152  000000  EMT341: .WORD   EMS55,EMS34,0
071356  073733  075244  074030  EMT342: .WORD   EMS52,EMS117,EMS56,EMS57,0
071370  073733  075244  073535  EMT343: .WORD   EMS52,EMS117,EMS45,EMS57,0
071402  073733  075244  073462  EMT344: .WORD   EMS52,EMS117,EMS44,EMS57,0
071414  073733  075244  100023  EMT345: .WORD   EMS52,EMS117,EMS221,0
071424  074770  072630  075207  EMT346: .WORD   EMS106,EMS20,EMS115,EMS223,EMS72,0
071440  073733  077500  100073  EMT347: .WORD   EMS52,EMS205,EMS222,EMS206,0
071452  074513  076405  074654  EMT350: .WORD   EMS75,EMS150,EMS102,EMS115,EMS56,EMS73,0
071470  077053  074457  074654  EMT351: .WORD   EMS167,EMS73,EMS102,0
071500  077677  000000          EMT352: .WORD   EMS215,0
071504  077750  077437  077720  EMT353: .WORD   EMS217,EMS203,EMS216,0
071514  100023  072630  000000  EMT354: .WORD   EMS221,EMS20,0
```

```
071522  100253  101067  101146  EMT1:    .WORD   EM1,STSM1,STSM2,STSM4,0
071534  101067  101146  101277  EMT2:    .WORD   STSM1,STSM2,STSM4,0

071544  100272  000000          EMT110:  .WORD   EM110,0
071550  100301  000000          EMT111:  .WORD   EM111,0

071554  100320  000000          EMT114:  .WORD   EM114,0
071560  100347  101067  101146  EMT223:  .WORD   EM223,STSM1,STSM2,STSM4,0
071572  100375  101067  101146  EMT256:  .WORD   EM256,STSM1,STSM2,STSM4,0

071604  100452  101067  101146  EMT336:  .WORD   EM336,STSM1,STSM2,STSM4,0
071616  100511  101067  101146  EMT337:  .WORD   EM337,STSM1,STSM2,STSM4,0
071630  100650  101067  101146  EMT344:  .WORD   EM344,STSM1,STSM2,STSM4,0

071642  101010  000000          EMT353:  .WORD   EM353,0
```

```
071646  101340  101434  101452  EDT1:   .WORD   ED1,STSD1,STSD2,STSD4
071656  101434  101452  101504  EDT2:   .WORD   STSD1,STSD2,STSD4

071664  101346                  EDT110: .WORD   ED110
071666  101352                  EDT111: .WORD   ED111

071670  101360                  EDT114: .WORD   ED114
071672  101370  101434  101452  EDT223: .WORD   ED223,STSD1,STSD2,STSD4

071702  101400  101434  101452  EDT336: .WORD   ED336,STSD1,STSD2,STSD4
071712  101412  101434  101452  EDT337: .WORD   ED337,STSD1,STSD2,STSD4
071722  101412  101434  101452  EDT344: .WORD   ED337,STSD1,STSD2,STSD4,0

071734  101424                  EDT353: .WORD   ED353
```

```
071736  101517  101535  101535  EFT1:   .WORD   EF111,STSF,STSF,STSF
071746  101535  101535  101535  EFT2:   .WORD   STSF,STSF,STSF

071754  101516                  EFT110: .WORD   EF110
071756  101517                  EFT111: .WORD   EF111

071760  101521                  EFT114: .WORD   EF114
071762  101521  101535  101535  EFT223: .WORD   EF114,STSF,STSF,STSF

071772  101524  101535  101535  EFT336: .WORD   EF336,STSF,STSF,STSF
072002  101524  101535  101535  EFT337: .WORD   EF336,STSF,STSF,STSF
072012  101524  101535  101535  EFT344: .WORD   EF336,STSF,STSF,STSF

072022  101521                  EFT353: .WORD   EF114
```

```
                                .SBTTL  ERROR MESSAGE STRINGS

072024 051127 047117 020107 EMS1:   .ASCIZ  @WRONG UNIT SELECTED (RMCS2, BITS 0-2) @
072073    104 053105 041511 EMS2:   .ASCIZ  @DEVICE WENT @
072110 047125 053101 044501 EMS3:   .ASCIZ  @UNAVAILABLE "DVA" (RMCS1, BIT 11) @
072153    116 047117 054105 EMS4:   .ASCIZ  @NONEXISTENT "NED" (RMCS2, BIT 12) @
072216 047503 046515 047101 EMS5:   .ASCIZ  @COMMAND NOT COMPLETED, @
072246 047503 052116 047522 EMS6:   .ASCIZ  @CONTROLLER NOT READY (RMCS1, BIT 7) @
072313    104 044522 042526 EMS7:   .ASCIZ  @DRIVE NOT READY "DRY" (RMDS, BIT 7) @
072360 047507 047040 052117 EMS10:  .ASCIZ  @GO NOT RESET "GO" (RMCS1, BIT 0) @
072422 047111 040526 044514 EMS11:  .ASCIZ  @INVALID @
072433    106 047125 052103 EMS12:  .ASCIZ  @FUNCTION CODE (RMCS1, BITS 1-5) @
072474 040515 051523 052502 EMS13:  .ASCIZ  @MASSBUS @
072505    103 047117 051124 EMS14:  .ASCIZ  @CONTROL @
072516 052502 020123 040520 EMS15:  .ASCIZ  @BUS PARITY ERROR @
072540 046442 050103 021105 EMS16:  .ASCIZ  @"MCPE" (RMCS1, BIT 13) @
072570 051124 047101 043123 EMS17:  .ASCIZ  @TRANSFER ERROR (RMCS1, BIT 14) @
072630 044123 052517 042114 EMS20:  .ASCIZ  @SHOULD NOT BE SET @
072653    127 051117 020104 EMS21:  .ASCIZ  @WORD COUNT (RMWC) @
072676 052502 020123 051050 EMS22:  .ASCIZ  @BUS (RMBA) @
072712 046042 052102 020042 EMS23:  .ASCIZ  @"LBT" (RMDS, BIT 10) @
072740 040442 042517 020042 EMS24:  .ASCIZ  @"AOE" (RMER1, BIT 09) @
072767    104 051511 020113 EMS25:  .ASCIZ  @DISK (RMDA) @
073004 054503 044514 042116 EMS26:  .ASCIZ  @CYLINDER (RMDC) @
073025    101 042104 042522 EMS27:  .ASCIZ  @ADDRESS @
073036 052123 052101 051525 EMS30:  .ASCIZ  @STATUS @
073046 053442 042514 020042 EMS31:  .ASCIZ  @"WLE" (RMER1, BIT 11) @
073075    042 050125 021105 EMS32:  .ASCIZ  @"UPE" (RMCS2, BIT 13) @
073124 053442 043103 020042 EMS33:  .ASCIZ  @"WCF" (RMER1, BIT 5) @
073152 051127 052111 020105 EMS34:  .ASCIZ  @WRITE CHECK ERROR-"WCE" (RMCS2, BIT 14) @
073223    042 042115 042520 EMS35:  .ASCIZ  @"MDPE" (RMCS2, BIT 8) @
073252 042042 045503 020042 EMS36:  .ASCIZ  @"DCK" (RMER1, BIT 15) @
073301    042 041505 021110 EMS37:  .ASCIZ  @"ECH" (RMER1, BIT 6) @
073327    042 046104 021124 EMS40:  .ASCIZ  @"DLT" (RMCS2, BIT 15) @
073356 046442 043130 020042 EMS41:  .ASCIZ  @"MXF" (RMCS2, BIT 9) @
073404 042042 042524 020042 EMS42:  .ASCIZ  @"DTE" (RMER1, BIT 12) @
073433    042 041510 041522 EMS43:  .ASCIZ  @"HCRC" (RMER1, BIT 8) @
073462 042510 042101 051105 EMS44:  .ASCIZ  @HEADER COMPARE ERROR "HCE" (RMER1, BIT 7)\ @
073535    106 051117 040515 EMS45:  .ASCIZ  @FORMAT ERROR "FER" (RMER1, BIT 4) @
073600 044442 042501 020042 EMS46:  .ASCIZ  @"IAE" (RMER1, BIT 10) @
073627    042 050117 021111 EMS47:  .ASCIZ  @"OPI" (RMER1, BIT 13) @
073656 051442 044513 020042 EMS50:  .ASCIZ  @"SKI" (RMER2, BIT 14) @
073705    042 044520 021120 EMS51:  .ASCIZ  @"PIP" (RMDS, BIT 13) @
073733    124 042510 051040 EMS52:  .ASCIZ  @THE RM03 @
073745    104 052105 041505 EMS53:  .ASCIZ  @DETECTED @
073757    101 020124 047101 EMS54:  .ASCIZ  @AT AN UNEXPECTED @
074001    111 041516 051117 EMS55:  .ASCIZ  @INCORRECT DATA DURING @
074030 047111 040526 044514 EMS56:  .ASCIZ  @INVALID COMMAND ERROR "IVC" (RMER2, BIT 12) @
074105    104 051125 047111 EMS57:  .ASCIZ  @DURING DATA TRANSFER @
074133    104 052101 020101 EMS60:  .ASCIZ  @DATA READ @
074146 047504 051505 047040 EMS61:  .ASCIZ  @DOES NOT COMPARE WITH @
074175    104 052101 020101 EMS62:  .ASCIZ  @DATA WRITTEN @
074213    042 040520 021122 EMS63:  .ASCIZ  @"PAR" (RMER1, BIT 3) @
074241    111 020123 047111 EMS64:  .ASCIZ  @IS INCORRECT @
074257    103 046517 047520 EMS65:  .ASCIZ  @COMPOSITE ERROR "ERR" (RMDS, BIT 14) @
074325    104 052101 020101 EMS66:  .ASCIZ  @DATA PARITY ERROR "DPE" (RMER2, BIT 3) @
```

```
074375     104  051125  047111  EMS67:  .ASCIZ  @DURING SEEK COMMAND @
074422  051511  051040  051505  EMS70:  .ASCIZ  @IS RESET @
074434  051105  047522  042516  EMS71:  .ASCIZ  @ERRONEOUS @
074447     111  020123  042523  EMS72:  .ASCIZ  @IS SET @
074457     104  042111  047040  EMS73:  .ASCIZ  @DID NOT SET @
074474  044504  020104  047516  EMS74:  .ASCIZ  @DID NOT RESET @
074513     114  051517  020124  EMS75:  .ASCIZ  @LOST @
074521     104  051125  047111  EMS76:  .ASCIZ  @DURING PACK ACK COMMAND @
074552  051042  051115  020042  EMS77:  .ASCIZ  @"RMR" (RMER1, BIT 2) @
074600  044442  051114  020042  EMS100: .ASCIZ  @"ILR" (RMER1, BIT 1) @
074626  044442  043114  020042  EMS101: .ASCIZ  @"ILF" (RMER1, BIT 0) @
074654  052504  044522  043516  EMS102: .ASCIZ  @DURING SEARCH COMMAND @
074703     105  051122  051117  EMS103: .ASCIZ  @ERROR @
074712  046042  041502  020042  EMS104: .ASCIZ  @"LBC" (RMER2, BIT 10) @
074741     042  051514  021103  EMS105: .ASCIZ  @"LSC" (RMER2, BIT 11) @
074770  042510  042101  051105  EMS106: .ASCIZ  @HEADER ERRORS @
075007     102  051525  052040  EMS110: .ASCIZ  @BUS TIMEOUT (04 TRAP) @
075036  042101  051104  051505  EMS111: .ASCIZ  @ADDRESS @
075047     127  042510  020116  EMS112: .ASCII  @WHEN READING/WRITING RH REGISTERS @
075111     101  020124  044124          .ASCIZ  @AT THE FOLLOWING @
075133     124  042510  051440  EMS113: .ASCIZ  @THE SELECTED DEVICE IS @
075163     116  047117  054105  EMS114: .ASCIZ  @NONEXISTENT DEVICE @
075207     040  006455  000012  EMS115: .ASCIZ  @ -@<CR><LF>
075214  044124  020105  040515  EMS116: .ASCIZ  @THE MASSBUS CONTROLLER @
075244  040506  046111  042105  EMS117: .ASCIZ  @FAILED TO DETECT @
075266  047516  020124  047101  EMS120: .ASCIZ  @NOT AN RM03 @
075303     103  047117  051124  EMS121: .ASCIZ  @CONTROL STATUS REGISTER 1, RMCS1, @
075346  052502  020123  042101  EMS122: .ASCIZ  @BUS ADDRESS REGISTER, RMBA, @
075403     103  047117  051124  EMS123: .ASCIZ  @CONTROL STATUS REGISTER 2, RMCS2, @
075446  051105  047522  020122  EMS124: .ASCIZ  @ERROR REGISTER 1, RMER1, @
075500  052101  042524  052116  EMS125: .ASCIZ  @ATTENTION SUMMARY REGISTER, RMAS, @
075543     115  044501  052116  EMS126: .ASCIZ  @MAINTENANCE REGISTER #1, RMMR #1, @
075606  041505  020103  047520  EMS127: .ASCIZ  @ECC POSITION REGISTER, RMEC1, @
075645     105  041503  050040  EMS130: .ASCIZ  @ECC PATTERN REGISTER, RMEC2, @
075703     115  044501  052116  EMS131: .ASCIZ  @MAINTENANCE REGISTER 2, RMMR2, @
075743     116  052117  044440  EMS132: .ASCIZ  @NOT INITIALIZED BY @
075767     125  044516  052502  EMS133: .ASCIZ  @UNIBUS INITIALIZE @
076012  047503  052116  047522  EMS134: .ASCIZ  @CONTROLLER CLEAR, I.E.  BIT 5 OF @
076054  044122  030461  042440  EMS135: .ASCIZ  @RH11 ERROR CLEAR (RMCS1, BIT 14) @
076116  051104  053111  020105  EMS136: .ASCIZ  @DRIVE CLEAR COMMAND @
076143     104  044522  042526  EMS137: .ASCIZ  @DRIVE STATUS REGISTER, RMDS @
076200  042515  044504  046525  EMS140: .ASCIZ  @MEDIUM OFF LINE @
076221     042  047515  021114  EMS141: .ASCIZ  @"MOL" (RMDS, BIT 12) @
076247     104  044522  042526  EMS142: .ASCIZ  @DRIVE FAULT @
076264  042042  041526  020042  EMS143: .ASCIZ  @"DVC" (RMER2, BIT 7) @
076312  047125  040523  042506  EMS144: .ASCIZ  @UNSAFE @
076322  052442  051516  020042  EMS145: .ASCIZ  @"UNS" (RMER1, BIT 14) @
076351     123  047510  046125  EMS146: .ASCIZ  @SHOULD BE SET @
076370  043117  051506  052105  EMS147: .ASCIZ  @OFFSET MODE @
076405     040  047526  052514  EMS150: .ASCIZ  @ VOLUME VALID @
076424  047442  021115  024040  EMS151: .ASCIZ  @"OM" (RMDS, BIT 0) @
076450  053042  021126  024040  EMS152: .ASCIZ  @"VV" (RMDS, BIT 6) @
076474  040520  045503  040440  EMS153: .ASCIZ  @PACK ACK COMMAND @
076516  047516  020124  042523  EMS154: .ASCIZ  @NOT SET BY @
076532  043117  051506  052105  EMS155: .ASCIZ  @OFFSET COMMAND @
076552  047516  020124  042522  EMS156: .ASCIZ  @NOT RESET BY @
```

```
076570 052122 020103 047503 EMS157: .ASCIZ  @RTC COMMAND @
076605    122 050111 041440 EMS160: .ASCIZ  @RIP COMMAND @
076622 043117 051506 052105 EMS161: .ASCIZ  @OFFSET REGISTER (RMOF) @
076652 051105 047522 020122 EMS162: .ASCIZ  @ERROR REGISTER #2, RMER2, @
076705    104 051125 047111 EMS163: .ASCIZ  @DURING RECALIBRATE @
076731    111 020123 047111 EMS164: .ASCII  @IS INTERMITTENT OR DRIVE DIDNT DROP ON @
077000 054503 044514 042116       .ASCIZ  @CYLINDER @
077012 047125 054105 042520 EMS165: .ASCIZ  @UNEXPECTED @
077026 042522 040503 044514 EMS166: .ASCIZ  @RECALIBRATE COMMAND @
077053    042 052101 021101 EMS167: .ASCIZ  @"ATA" (RMDS, BIT15) @
077100 044127 047105 051040 EMS170: .ASCIZ  @WHEN READING REGISTER @
077127    105 051122 051117 EMS171: .ASCIZ  @ERROR REGISTER #2, RMER2, @
077162 047516 051116 041505 EMS172: .ASCIZ  @NONRECOVERABLE @
077202 042522 047503 042526 EMS173: .ASCIZ  @RECOVERABLE @
077217    104 052101 020101 EMS174: .ASCIZ  @DATA @
077225    104 051125 047111 EMS175: .ASCIZ  @DURING WRITE COMMAND @
077253    042 050117 021105 EMS176: .ASCIZ  @"DPE" (RMER2, BIT 13) @
077302 047111 053440 044522 EMS177: .ASCIZ  @IN WRITE PROTECT @
077324 040503 020116 047516 EMS200: .ASCIZ  @CAN NOT SET @
077341    104 040511 047107 EMS201: .ASCIZ  @DIAGNOSTIC MODE "DMD" (RMMR1, BIT 0) @
077407    104 051125 047111 EMS202: .ASCIZ  @DURING DIAGNOSTIC MODE @
077437    111 041516 051117 EMS203: .ASCIZ  @INCORRECT @
077452 053442 046122 020042 EMS204: .ASCIZ  @"WRL" (RMDS, BIT 11) @
077500 054105 041505 052125 EMS205: .ASCIZ  @EXECUTED @
077512 044527 044124 041440 EMS206: .ASCIZ  @WITH COMP ERROR SET @
077537    042 047507 020042 EMS207: .ASCIZ  @"GO" (RMCS1, BIT 0) @
077564 051127 052111 047111 EMS210: .ASCIZ  @WRITING @
077575    127 051501 051040 EMS211: .ASCIZ  @WAS RESET BY @
077613    120 047522 051107 EMS212: .ASCIZ  @PROGRAM INTERRUPT @
077636 040527 020123 047516 EMS213: .ASCIZ  @WAS NOT GENERATED @
077661    123 042505 020113 EMS214: .ASCIZ  @SEEK COMMAND @
077677    120 047522 051107 EMS215: .ASCIZ  @PROGRAM TIMEOUT @
077720 052504 044522 043516 EMS216: .ASCIZ  @DURING LOOK AHEAD TEST @
077750 047514 045517 040440 EMS217: .ASCIZ  @LOOK AHEAD REGISTER,RMLA, @
100003    123 040505 041522 EMS220: .ASCIZ  @SEARCH COMMAND @
100023    102 042101 051440 EMS221: .ASCIZ  @BAD SECTOR ERROR "BSE" (RMER2, BIT 15) @
100073    101 042040 052101 EMS222: .ASCIZ  @A DATA TRANSFER COMMAND @
100124 042510 042101 051105 EMS223: .ASCIZ  @HEADER COMPARE INHIBIT "HCI" (RMOF, BIT 10) @
100201    116 047117 054105 EMS224: .ASCIZ  @NONEXISTENT MEMORY "NEM" (RMCS2, BIT 11) @
```

```
                                                    D 6
100253    105  050130  052103  EM1:    .ASCIZ  @EXPCTD  RECEVD@
100272 052502  040523  051104  EM110:  .ASCIZ  @BUSADR@
100301    040  046522  051503  EM111:  .ASCIZ  @ RMCS2   RMCS1@

100320 042522  042503  042126  EM114:  .ASCIZ  @RECEVD  SNGPRT  DULPRT@
100347    105  050130  052103  EM223:  .ASCIZ  @EXPCTD  RECEVD   DATA@
100375    105  050130  052103  EM256:  .ASCII  @EXPCTD  RECEVD  RGSTR@<CR><LF>
100424 052123  052101  051525          .ASCIZ  @STATUS  STATUS  INDEX@

100452 042107  042101  051522  EM336:  .ASCIZ  @GDADRS  GDDATA  BDADRS  BDDATA@
100511    122  041515  031123  EM337:  .ASCII  @RMCS2   STATUS  FAILING   DATA@<CR><LF>
100551    137  057537  057537          .ASCII  @                          @<CR><LF>
100611    105  050130  052103          .ASCIZ  @EXPCTD--RECEVD  --BIT---ADRESS@

100650 046522  051105  020061  EM344:  .ASCII  @RMER1   STATUS  HEADER  FAILING@<CR><LF>
100711    137  057537  057537          .ASCII  @                  WORD     BIT@<CR><LF>
100750 054105  041520  042124          .ASCIZ  @EXPCTD--RECEVD  NUMBER  POSITON@
101010 054105  041520  042124  EM353:  .ASCII  @EXPCTD  RECEVD@<CR><LF>
101030 041474  037122  046074          .ASCIZ  @<CR><LF> RMLA    RMLA    RMOF @

101067    040  046522  051503  STSH1:  .ASCII  @ RMCS1  RMCS2   RMDS    RMER1   RMER2@
101134 020040  051040  040515          .ASCIZ  @   RMAS@<CR><LF>
101146 051040  053515  020103  STSH2:  .ASCII  @ RMWC    RMBA    RMDA    RMOF    RMDC@
101213    040  020040  051040          .ASCIZ  @    RMEC1   RMEC2@<CR><LF>
101237    040  046522  040504  STSH3:  .ASCIZ  @ RMDA    RMDC    RMOF    RMLA@<CR><LF>
101277    040  046522  051115  STSH4:  .ASCIZ  @ RMMR1   RMMR2   RMDT    RMSN@<CR><LF>
```

```
                101340                          .EVEN
101340  001140  001142  000000  ED1:    .WORD   $GDDAT,$BDDAT,0
101346  001276  000000          ED110:  .WORD   $BASE,0
101352  001174  001176  000000  ED111:  .WORD   $TMP0,$TMP1,0

101360  001356  001176  001200  ED114:  .WORD   RMDTI,$TMP1,$TMP2,0
101370  001140  001142  001174  ED223:  .WORD   $GDDAT,$BDDAT,$TMP0,0

101400  001134  001140  001136  ED336:  .WORD   $GDADR,$GDDAT,$BDADR,$BDDAT,0

101412  001140  001142  001174  ED337:  .WORD   $GDDAT,$BDDAT,$TMP0,$TMP1,0
101424  001140  001142  001432  ED353:  .WORD   $GDDAT,$BDDAT,RMOF0,0

101434  001330  001340  001342  STSD1:  .WORD   RMCS1I,RMCS2I,RMDSI,RMER1I,RMER2I,RMASI,0
101452  001332  001334  001336  STSD2:  .WORD   RMWCI,RMBAI,RMDAI,RMOFI,RMDCI,RMEC1I
101466  001376  000000          .WORD   RMEC2I,0
101472  001336  001364  001362  STSD3:  .WORD   RMDAI,RMDCI,RMOFI,RMLAI,0
101504  001354  001370  001356  STSD4:  .WORD   RMMR1I,RMMR2I,RMDTI,RMSNI,0
```

```
101516  000                     EF110:  .BYTE   0
101517  000     000             EF111:  .BYTE   0,0
101521  000     000     000     EF114:  .BYTE   0,0,0
101524  000     000     000     EF336:  .BYTE   0,0,0,0
101530  000     000     000     EF337:  .BYTE   0,0,0,0,0

101535  000     000     000     STSF:   .BYTE   0,0,0,0,0,0,0
```

```
                                .EVEN
101544  000402          MFGFIL: .BLKW   258.
102550  177777                  .WORD   -1
102552  177777                  .WORD   -1
102554  000402          USRFIL: .BLKW   258.
103560  177777                  .WORD   -1
103562  177777                  .WORD   -1
                                .EVEN
```

```
                                        H 6
103564                          BUFFER:
103564  000402                  BUFONE: .BLKW   258.
104570  000402                  BUFTWO: .BLKW   258.


        103564                  .       =       BUFFER
103564                          HELP:
103564  005015                  .ASCII  <CR><LF>
                                ;.ASCII @CAUTION: PARTS 2 AND 3 OF THE FUNCTIONAL TEST LEAVE @<CR><LF>
                                ;.ASCII @BAD HEADERS ON THE PACK SO BE SURE TO FORMAT THE PACKS@<CR><LF>
                                ;.ASCII @WHEN DONE.@<CR><LF>
103566  046011  051511  020124  .ASCII  @       LIST OF TESTS@<CR><LF>
103606  057411  057537  057537  .ASCII  @                       @<CR><LF>
103626  030524  041411  047117  .ASCII  @T1     CONTROLLER ACCESS TEST@<CR><LF>
103661     124  004462  042504  .ASCII  @T2     DEVICE AVAILABLE TEST@<CR><LF>
103713     124  004463  051104  .ASCII  @T3     DRIVE TYPE TEST@<CR><LF>
103737     124  004464  047506  .ASCII  @T4     FORMAT ZEROS - 18@<CR><LF>
103765     124  004465  047506  .ASCII  @T5     FORMAT ZEROS - 16@<CR><LF>
104013     124  004466  042532  .ASCII  @T6     ZERO FILL TEST@<CR><LF>
104036  033524  043011  051117  .ASCII  @T7     FORMAT CHECK ZEROS - 16@<CR><LF>
104072  030524  004460  047506  .ASCII  @T10    FORMAT CHECK ZEROS W/ WCE ERROR@<CR><LF>
104137     124  030461  043011  .ASCII  @T11    FORMAT ONES - 16@<CR><LF>
104165     124  031061  043011  .ASCII  @T12    FORMAT CHECK ONES - 16@<CR><LF>
104221     124  031461  043011  .ASCII  @T13    FORMAT CHECK ONES W/ WCE ERRORS@<CR><LF>
104266  030524  004464  047506  .ASCII  @T14    FORMAT MULTIPLE SECTORS@<CR><LF>
104323     124  032461  043011  .ASCII  @T15    FORMAT W/ HEAD SWITCHING@<CR><LF>
104361     124  033061  043011  .ASCII  @T16    FORMAT W/ MID TRANSFER SEEK@<CR><LF>
104422  030524  004467  047506  .ASCII  @T17    FORMAT W/ IMPLIED SEEK@<CR><LF>
104456  031124  004460  047506  .ASCII  @T20    FORMAT EACH SECTOR ADDRESS@<CR><LF>
104516  031124  004461  047506  .ASCII  @T21    FORMAT EACH TRACK ADDRESS@<CR><LF>
104555     124  031062  043011  .ASCII  @T22    FORMAT PRIME CYLINDERS@<CR><LF>
104611     124  031462  043011  .ASCII  @T23    FORMAT LAST SECTOR@<CR><LF>
104641     124  032062  043011  .ASCII  @T24    FORMAT W/ AOE ERROR@<CR><LF>
104672  031124  004465  047506  .ASCII  @T25    FORMAT INVALID SECTOR ADDRESS@<CR><LF>
104735     124  033062  043011  .ASCII  @T26    FORMAT INVALID TRACK ADDRESS@<CR><LF>
104777     124  033462  043011  .ASCII  @T27    FORMAT INVALID CYLINDER ADDRESS@<CR><LF>
105044  031524  004460  047506  .ASCII  @T30    FORMAT AT OFFSET@<CR><LF>
105072  031524  004461  053111  .ASCII  @T31    IVC FORMAT TEST@<CR><LF>
105117     124  031063  043011  .ASCII  @T32    FORMAT ERROR TEST - 18@<CR><LF>
105153     124  031463  043011  .ASCII  @T33    FORMAT ERROR TEST - 16@<CR><LF>
105207     124  032063  043011  .ASCII  @T34    FORMAT HCE, FIRST HEADER WORD@<CR><LF>
105252  031524  004465  047506  .ASCII  @T35    FORMAT HCE, SECOND HEADER WORD@<CR><LF>
105316  005015                  .ASCII  <CR><LF>
105320  047411  042520  040522  .ASCII  @       OPERATIONAL SWITCH SETTINGS@<CR><LF>
105356  057411  057537  057537  .ASCII  @       ---------------------------@<CR><LF>
105414  005015                  .ASCII  <CR><LF>
105416  053523  052111  044103  .ASCII  @SWITCH                  USE@<CR><LF>
105434  026455  026455  026455  .ASCII  @------                  ----------@<CR><LF>
105472  020040  032461  004411  .ASCII  @  15                    HALT ON ERROR@<CR><LF>
105517     040  030440  004464  .ASCII  @  14                    LOOP ON TEST@<CR><LF>
105543     040  030440  004463  .ASCII  @  13                    INHIBIT ERROR TYPEOUTS@<CR><LF>
105601     040  030440  004462  .ASCII  @  12                    @<CR><LF>
105611     040  030440  004461  .ASCII  @  11                    INHIBIT ITERATIONS@<CR><LF>
105643     040  030440  004460  .ASCII  @  10                    BELL ON ERROR@<CR><LF>
105670  020040  034440  004411  .ASCII  @   9                    LOOP ON ERROR@<CR><LF>
105715     040  020040  004470  .ASCII  @   8                    LOOP ON TEST IN SWR<7:0>@<CR><LF>
```

```
105755     040  020040  004467  .ASCII  @   7          TN128@<CR><LF>
105772  020040  033040  004411  .ASCII  @   6          TN64@<CR><LF>
106006  020040  032440  004411  .ASCII  @   5          TN32@<CR><LF>
106022  020040  032040  004411  .ASCII  @   4          TN16@<CR><LF>
106036  020040  031440  004411  .ASCII  @   3          TN8@<CR><LF>
106051     040  020040  004462  .ASCII  @   2          TN4@<CR><LF>
106064  020040  030440  004411  .ASCII  @   1          TN2@<CR><LF>
106077     040  020040  004460  .ASCII  @   0          TN1@<CR><LF>
106112  005015                  .ASCII  <CR><LF>
106114  005015     000          .ASCIZ  <CR><LF>
        000001                  .END
```

```
ARGS   = 000004   3672# 3709# 3741# 3748# 3755# 3784# 3791# 3798# 3807# 3840# 3877# 3909# 3916#
                  3923# 3952# 3959# 3966# 3975# 4008# 4045# 4078# 4085# 4092# 4122# 4129# 4136#
                  4145# 4178# 4215# 4247# 4254# 4261# 4289# 4296# 4303# 4338# 4375# 4407# 4414#
                  4421# 4452# 4463# 4526# 4563# 4595# 4602# 4609# 4638# 4645# 4652# 4661# 4694#
                  4731# 4763# 4770# 4777# 4805# 4812# 4819# 4854# 4891# 4923# 4930# 4937# 4968#
                  4979# 5042# 5079# 5111# 5118# 5125# 5153# 5160# 5167# 5201# 5238# 5270# 5277#
                  5284# 5312# 5319# 5326# 5360# 5397# 5429# 5436# 5443# 5471# 5478# 5485# 5519#
                  5558# 5591# 5598# 5605# 5633# 5640# 5647# 5684# 5722# 5729# 5736# 5765# 5772#
                  5779# 5788# 5829# 5867# 5874# 5881# 5910# 5917# 5924# 5933# 5974# 6012# 6019#
                  6026# 6055# 6062# 6069# 6078# 6107# 6155# 6162# 6169# 6191# 6237# 6244# 6251#
                  6286# 6324# 6331# 6338# 6347# 6378# 6385# 6392# 6433# 6471# 6478# 6485# 6494#
                  6525# 6532# 6539# 6580# 6618# 6625# 6632# 6641# 6672# 6679# 6686# 6726# 6776#
                  6783# 6790# 6838# 6845# 6852# 6861# 6901# 6952# 6966# 6975# 7006# 7020# 7054#
                  7092# 7099# 7106# 7136# 7147# 7158# 7192# 7230# 7237# 7244# 7274# 7285# 7296#
                  7331# 7378# 7385# 7392# 7421# 7436# 7451# 7471# 7489# 7516# 7567# 7614# 7621#
                  7628# 7657# 7668# 7686# 7711#
ASNDA  001500     1487# 8464* 8493  8539* 8540  8542* 8543* 8544  8546* 8591
ASNDC  001476     1486# 8463* 8492  8547* 8548  8550* 8590
ASWREG= 000000    1363  1376
ATA    = 100000   1011# 9326  9327  9329  10061 10062 10097 10100 10592 10593 10632 10635 12557
ATESTN= 000000    1363  1367
ATNMSK= 000377    1048# 10789
ATNTBL  064772    3469  3487  12557#
AUNIT  = 000000   1363  1370
AUSWR  = 000000   1363  1377
AVECT1= 120254    1252# 1363  1402
AVECT2= 000000    1363  1403
A16    = 000400   1201#
A17    = 001000   1200#
BACK   = 000000   3672# 3677  3709# 3713  3741# 3745  3748# 3752  3755# 3759  3784# 3788  3791#
                  3795  3798# 3802  3807# 3813  3840# 3845  3877# 3881  3909# 3913  3916# 3920
                  3923# 3927  3952# 3956  3959# 3963  3966# 3970  3975# 3981  4008# 4013  4045#
                  4049  4078# 4082  4085# 4089  4092# 4096  4122# 4126  4129# 4133  4136# 4140
                  4145# 4151  4178# 4183  4215# 4219  4247# 4251  4254# 4258  4261# 4265  4289#
                  4293  4296# 4300  4303# 4307  4338# 4343  4375# 4379  4407# 4411  4414# 4418
                  4421# 4425  4452# 4456  4463# 4467  4526# 4531  4563# 4567  4595# 4599  4602#
                  4606  4609# 4613  4638# 4642  4645# 4649  4652# 4656  4661# 4667  4694# 4699
                  4731# 4735  4763# 4767  4770# 4774  4777# 4781  4805# 4809  4812# 4816  4819#
                  4823  4854# 4859  4891# 4895  4923# 4927  4930# 4934  4937# 4941  4968# 4972
                  4979# 4983  5042# 5047  5079# 5083  5111# 5115  5118# 5122  5125# 5129  5153#
                  5157  5160# 5164  5167# 5171  5201# 5206  5238# 5242  5270# 5274  5277# 5281
                  5284# 5288  5312# 5316  5319# 5323  5326# 5330  5360# 5365  5397# 5401  5429#
                  5433  5436# 5440  5443# 5447  5471# 5475  5478# 5482  5485# 5489  5519# 5524
                  5558# 5562  5591# 5595  5598# 5602  5605# 5609  5633# 5637  5640# 5644  5647#
                  5651  5684# 5689  5722# 5726  5729# 5733  5736# 5740  5765# 5769  5772# 5776
                  5779# 5783  5788# 5794  5829# 5834  5867# 5871  5874# 5878  5881# 5885  5910#
                  5914  5917# 5921  5924# 5928  5933# 5939  5974# 5979  6012# 6016  6019# 6023
                  6026# 6030  6055# 6059  6062# 6066  6069# 6073  6078# 6084  6107# 6112  6155#
                  6159  6162# 6166  6169# 6173  6191# 6196  6237# 6241  6244# 6248  6251# 6255
                  6286# 6291  6324# 6328  6331# 6335  6338# 6342  6347# 6352  6378# 6382  6385#
                  6389  6392# 6396  6433# 6438  6471# 6475  6478# 6482  6485# 6489  6494# 6499
                  6525# 6529  6532# 6536  6539# 6543  6580# 6585  6618# 6622  6625# 6629  6632#
                  6636  6641# 6646  6672# 6676  6679# 6683  6686# 6690  6726# 6731  6776# 6780
                  6783# 6787  6790# 6794  6838# 6842  6845# 6849  6852# 6856  6861# 6867  6901#
                  6906  6952# 6956  6966# 6970  6975# 6980  7006# 7010  7020# 7024  7054# 7059
                  7092# 7096  7099# 7103  7106# 7110  7136# 7140  7147# 7151  7158# 7162  7192#
```

```
                    7197  7230#  7234  7237#  7241  7244#  7248  7274#  7278  7285#  7289  7296#  7300
                    7331# 7336   7378# 7382   7385# 7389   7392# 7396   7421# 7425   7436# 7440   7451#
                    7455  7471#  7475  7489#  7493  7516#  7521  7567#  7572  7614#  7618  7621#  7625
                    7628# 7632   7657# 7661   7668# 7672   7686# 7690   7711# 7716
BADSCT  034714      8180#
BA1    = 000010     1220#  9715
BB00   = 000001     1138#
BB01   = 000002     1137#
BB02   = 000004     1136#
BB03   = 000010     1135#
BB04   = 000020     1134#
BB05   = 000040     1133#
BB06   = 000100     1132#
BB07   = 000200     1131#
BB08   = 000400     1130#
BB09   = 001000     1129#
BIT0   = 000001      927#  3316   3637  3639   9002
BIT00  = 000001      917#   927    953  1002   1021  1040  1078  1097  1138  1224  1240
BIT01  = 000002      916#   926    952  1001   1039  1077  1096  1137  1223  1239
BIT02  = 000004      915#   925    951  1000   1038  1076  1095  1136  1222  1238
BIT03  = 000010      914#   924    950   999   1037  1075  1094  1135  1150  1220  1237
BIT04  = 000020      913#   923    949   998   1036  1093  1134  1219
BIT05  = 000040      912#   922    948  1035   1074  1092  1133  1218
BIT06  = 000100      911#   921   1020  1034   1056  1073  1091  1132  1203  1217  1236
BIT07  = 000200      910#   920   1019  1033   1055  1072  1090  1114  1131  1149  1202  1216
BIT08  = 000400      909#   919    997  1018   1032  1054  1071  1089  1130  1201  1214  11984
BIT09  = 001000      908#   918    996  1017   1031  1053  1070  1088  1129  1200  1213  12002  12100
BIT1   = 000002      926#  8470   8628  9732
BIT10  = 002000      907#   995   1016  1030   1052  1069  1087  1113  1128  1148  1199  1212  1235
                    12077
BIT11  = 004000      906#   947   1015  1029   1068  1086  1104  1112  1127  1147  1211  1234  8098
                    12009
BIT12  = 010000      905#  1014   1028  1067   1085  1111  1126  1146  1210  1233  8048
BIT13  = 020000      904#  1013   1027  1066   1084  1103  1125  1145  1197  1209  1232  12084
BIT14  = 040000      903#  1012   1026  1065   1083  1102  1124  1144  1155  1196  1208  1231  8013
                    11970
BIT15  = 100000      902#  1011   1025  1064   1082  1101  1123  1143  1154  1195  1207  1230  8000
                     8713  8767
BIT2   = 000004      925#  9002
BIT3   = 000010      924#  8126  11174  11261
BIT4   = 000020      923#  8076  11280
BIT5   = 000040      922#
BIT6   = 000100      921#  8025
BIT7   = 000200      920#  3311   9008
BIT8   = 000400      919#
BIT9   = 001000      918#
BOTADR  033736      7864* 7882*  7885  7900   7945#
BOTFLG  033740      7850* 7892*  7895  7898*  7946#
BPTVEC= 000014       934#
BSE    = 100000     1143#  7449  7460  8376   9619
BUFFER  103564      8195* 8444  12557#
BUFONE  103564      3664  3808   3832  3976   4000  4146  4170  4330  4518  4662  4686  4846  5034
                    5193  5352   5511  5676   5789  5821  5934  5966  6079  6278  6425  6572  6718
                    6862  6872   6893  7046   7184  7323  7340  7342* 7344* 7461  7480  7512  7559
                    7576  7578*  7580* 7677   7708 12557#
BUFTWO  104570      3766  3809   3934  3977   4104  4147  4430* 4477  4490  4620  4663  4946* 4992
```

```
                    5005    5747    5790    5892    5935    6037    6080    6120    6204    6357    6504    6651    6815
                    6863    6985    7118    7256    7403    7462    7481    7639    7678   12557#
CC     = 004000     1127#
CH     = 002000     1128#
CHRCNT  033741      7851*   7869*   7875*   7876    7879*   7883    7888*   7899*   7947#
CKSWR  = 104410    11968   12073   12099   12450#
CLEAN  = ****** U   7465   12557
CLKADR  001502      1488#   8964*   8966    8970*   8972    8997
CLKVCT  001504      1489#   8965*   8971*
CLR    = 000040     1218#   3318   10158
CLRSTS  046776      8035   10194#
CLSPRN  063570      3389    3405    3426   12557#
CMNSTA  006772      3341    3476#
CMPBUF  037064      3807    3975    4145    4661    5788    5933    6078    6861    8693#
CMPERR  057122     11573#
CNSL00  064067      3351   12557#
CNSL01  064126      3386   12557#
CNSL02  064153      3395   12557#
CNSL03  064234      3400   12557#
CNSL04  064264      3411   12557#
CNSL05  064340      3421   12557#
CNSL06  064374      3432   12557#
CNSL07  064421      3445   12557#
CNTCLR  046660      3576    3620    8015   10150#
CONT   = 000100     1091#
CR     = 000015      842#   3459    7867   11656   11943   11953   12557
CRLF   = 000200      843#   3295   11914   11953
CTLFG   001326      1418#   3515*  11645*  11647
CYLMSK= 001777      1118#   8631
DBCK   = 100000     1064#
DBEN   = 040000     1065#
DBL    = 002000     1235#
DCK    = 100000     1025#   1042    8369    8701    9657    9677   11267   11270
DDISP  = 177570      849#   1335    3259
DEBL   = 020000     1066#
DEVSEL  045206      8002   9860#
DISPLA  001156      1335#   3259*   3267*  12024*  12076*
DISPRE  000174      1262#   3267
DLT    = 100000     1207#   9501   11346   11349
DMD    = 000001     1078#   1097#   6910
DPE    = 000010     1150#   8370    8376    9163    9443    9939   10478   10899   11320   11323
DPEMI  = 040000     1231#
DPELO  = 020000     1232#
DPR    = 000400     1018#  10100   10297   10298   10764
DRQ    = 004000     1104#
DRVCLR= 000010       962#   8223   10752
DRVSTS  052214      8241   10743#
DRY    = 000200     1019#   9130    9226    9229    9230   10100   10297   10298   10764
DSWR   = 177570      848#   1334    3258
DTASTS  053016      3748    3791    3916    3959    4085    4129    4254    4296    4414    4463    4602    4645    4770
                    4812    4930    4979    5118    5160    5277    5319    5436    5478    5598    5640    5729    5772
                    5874    5917    6019    6062    6162    6244    6331    6385    6478    6532    6625    6679    6783
                    6845    7099    7147    7237    7285    7385    7436    7451    7471    7621    7668    8344   10874#
DTE    = 010000     1028#   1042   11133   11151   11154
DTO    = 010000     1067#
DULPRT= 024024      1107#   3635    3639    3642
```

```
DVA    = 004000          947#  3324  3595  8853  8926  9091  9094  9885 10203 10751 10752
DVC    = 000200         1149# 10032 10287 10539 10577 10580 10713 11086 11090 11093 11138 11512 11517
                        11520
EARLY    065316         12557#
EBL    = 020000          1084#
ECM    = 000100          1034#  1042  8369  8703  9640  9655  9673  9679 11277 12557
EC1    = 004000          1112#  8705  9675 11274
ECRC   = 001000          1088#
EDT1     071646          1523  1530  1537  1544  1551  1558  1572  1579  1586  1593  1600  1607  1614
                         1621  1628  1635  1642  1649  1656  1663  1670  1677  1684  1691  1698  1705
                         1712  1719  1726  1733  1740  1747  1754  1761  1768  1775  1782  1789  1796
                         1803  1810  1818  1825  1832  1839  1846  1854  1862  1870  1884  1891  1898
                         1905  1912  1919  1926  1934  1941  1948  1955  1962  1969  1976  1983  1990
                         1997  2004  2011  2018  2060  2067  2074  2081  2088  2095  2102  2109  2116
                         2123  2130  2137  2144  2151  2158  2165  2172  2179  2186  2193  2200  2207
                         2214  2221  2228  2235  2242  2249  2256  2263  2270  2277  2284  2291  2298
                         2305  2312  2319  2334  2341  2348  2355  2369  2376  2383  2390  2397  2404
                         2411  2418  2425  2432  2439  2446  2453  2460  2467  2474  2481  2488  2495
                         2516  2523  2530  2537  2544  2684  2691  2698  2719  2726  2733  2747  2754
                         2761  2768  2775  2782  2789  2796  2804  2811  2819  2826  2833  2841  2849
                         2857  2866  2874  2882  2889  2896  2903  2910  2917  2924  2931  2938  2945
                         2952  2959  2966  2973  2980  2987  2994  3001  3008  3015  3022  3029  3036
                         3043  3050  3057  3064  3071  3078  3113  3120  3134  3141  3148  3155  3162
                         3183 12557#
EDT110   071664         2025 12557#
EDT111   071666         2032  2039 12557#
EDT114   071670         2053 12557#
EDT2     071656         2502  2509  2705  2712 12557#
EDT223   071672         2551  2740 12557#
EDT336   071702         2327  3085  3099  3106 12557#
EDT337   071712         3092 12557#
EDT344   071722         3127 12557#
EDT353   071734         3176 12557#
ED1      101340        12557#
ED110    101346        12557#
ED111    101352        12557#
ED114    101360        12557#
ED223    101370        12557#
ED336    101400        12557#
ED337    101412        12557#
ED353    101424        12557#
EECC   = 000020          1093#
EFT1     071736          1524  1531  1538  1545  1552  1559  1573  1580  1587  1594  1601  1608  1615
                         1622  1629  1636  1643  1650  1657  1664  1671  1678  1685  1692  1699  1706
                         1713  1720  1727  1734  1741  1748  1755  1762  1769  1776  1783  1790  1797
                         1804  1811  1819  1826  1833  1840  1847  1355  1863  1871  1885  1892  1899
                         1906  1913  1920  1927  1935  1942  1949  1956  1963  1970  1977  1984  1991
                         1998  2005  2012  2019  2061  2068  2075  2082  2089  2096  2103  2110  2117
                         2124  2131  2138  2145  2152  2159  2166  2173  2180  2187  2194  2201  2208
                         2215  2222  2229  2236  2243  2250  2257  2264  2271  2278  2285  2292  2299
                         2306  2313  2320  2335  2342  2349  2356  2370  2377  2384  2391  2398  2405
                         2412  2419  2426  2433  2440  2447  2454  2461  2468  2475  2482  2489  2496
                         2517  2524  2531  2538  2545  2685  2692  2699  2720  2727  2734  2748  2755
                         2762  2769  2776  2783  2790  2797  2805  2812  2820  2827  2834  2842  2850
                         2858  2867  2875  2883  2890  2897  2904  2911  2918  2925  2932  2939  2946
                         2953  2960  2967  2974  2981  2988  2995  3002  3009  3016  3023  3030  3037
```

```
                    3044    3051    3058    3065    3072    3079    3114    3121    3135    3142    3149    3156    3163
                    3184   12557#
EFT110  071754      2026   12557#
EFT111  071756      2033    2040   12557#
EFT114  071760      2054   12557#
EFT2    071746      2503    2510    2706    2713   12557#
EFT223  071762      2552    2741   12557#
EFT336  071772      2328    3086    3100    3107   12557#
EFT337  072002      3093   12557#
EFT344  072012      3128   12557#
EFT353  072022      3177   12557#
EF110   101516     12557#
EF111   101517     12557#
EF114   101521     12557#
EF336   101524     12557#
EF337   101530     12557#
EMT1    071522      1522    1529    1536    1543    1550    1557    1571    1578    1585    1592    1599    1606    1613
                    1620    1627    1634    1641    1648    1655    1662    1669    1676    1683    1690    1697    1704
                    1711    1718    1725    1732    1739    1746    1753    1760    1767    1774    1781    1788    1795
                    1802    1809    1817    1824    1831    1838    1845    1853    1861    1869    1883    1890    1897
                    1904    1911    1918    1925    1933    1940    1947    1954    1961    1968    1975    1982    1989
                    1996    2003    2010    2017    2059    2066    2073    2080    2087    2094    2101    2108    2115
                    2122    2129    2136    2143    2150    2157    2164    2171    2178    2185    2192    2199    2206
                    2213    2220    2227    2234    2241    2248    2255    2262    2269    2276    2283    2290    2297
                    2304    2311    2318    2333    2340    2347    2354    2368    2375    2382    2389    2396    2403
                    2410    2417    2424    2431    2438    2445    2452    2459    2466    2473    2480    2487    2494
                    2515    2522    2529    2536    2543    2683    2690    2697    2718    2725    2732    2746    2753
                    2760    2767    2774    2781    2788    2795    2803    2810    2818    2825    2832    2840    2848
                    2856    2865    2873    2881    2888    2895    2902    2909    2916    2923    2930    2937    2944
                    2951    2958    2965    2972    2979    2986    2993    3000    3007    3014    3021    3028    3035
                    3042    3049    3056    3063    3070    3077    3112    3119    3133    3140    3147    3154    3161
                    3182   12557#
EMT110  071544      2024   12557#
EMT111  071550      2031    2038   12557#
EMT114  071554      2052   12557#
EMT2    071534      2501    2508    2704    2711   12557#
EMT223  071560      2550   12557#
EMT256  071572      2739   12557#
EMT336  071604      2326    3084    3098    3105   12557#
EMT337  071616      3091   12557#
EMT344  071630      3126   12557#
EMT353  071642      3175   12557#
EM1     100253     12557#
EM110   100272     12557#
EM111   100301     12557#
EM114   100320     12557#
EM223   100347     12557#
EM256   100375     12557#
EM336   100452     12557#
EM337   100511     12557#
EM344   100650     12557#
EM353   101010     12557#
EMS1    072024     12557#
EMS10   072360     12557#
EMS100  074600     12557#
EMS101  074626     12557#
```

EMS102 074654 12557#
EMS103 074703 12557#
EMS104 074712 12557#
EMS105 074741 12557#
EMS106 074770 12557#
EMS11 072422 12557#
EMS110 075007 12557#
EMS111 075036 12557#
EMS112 075047 12557#
EMS113 075133 12557#
EMS114 075163 12557#
EMS115 075207 12557#
EMS116 075214 12557#
EMS117 075244 12557#
EMS12 072433 12557#
EMS120 075266 12557#
EMS121 075303 12557#
EMS122 075346 12557#
EMS123 075403 12557#
EMS124 075446 12557#
EMS125 075500 12557#
EMS126 075543 12557#
EMS127 075606 12557#
EMS13 072474 12557#
EMS130 075645 12557#
EMS131 075703 12557#
EMS132 075743 12557#
EMS133 075767 12557#
EMS134 076012 12557#
EMS135 076054 12557#
EMS136 076116 12557#
EMS137 076143 12557#
EMS14 072505 12557#
EMS140 076200 12557#
EMS141 076221 12557#
EMS142 076247 12557#
EMS143 076264 12557#
EMS144 076312 12557#
EMS145 076322 12557#
EMS146 076351 12557#
EMS147 076370 12557#
EMS15 072516 12557#
EMS150 076405 12557#
EMS151 076424 12557#
EMS152 076450 12557#
EMS153 076474 12557#
EMS154 076516 12557#
EMS155 076532 12557#
EMS156 076552 12557#
EMS157 076570 12557#
EMS16 072540 12557#
EMS160 076605 12557#
EMS161 076622 12557#
EMS162 076652 12557#
EMS163 076705 12557#
EMS164 076731 12557#

EMS165	077012	12557#
EMS166	077026	12557#
EMS167	077053	12557#
EMS17	072570	12557#
EMS170	077100	12557#
EMS171	077127	12557#
EMS172	077162	12557#
EMS173	077202	12557#
EMS174	077217	12557#
EMS175	077225	12557#
EMS176	077253	12557#
EMS177	077302	12557#
EMS2	072073	12557#
EMS20	072630	12557#
EMS200	077324	12557#
EMS201	077341	12557#
EMS202	077407	12557#
EMS203	077437	12557#
EMS204	077452	12557#
EMS205	077500	12557#
EMS206	077512	12557#
EMS207	077537	12557#
EMS21	072653	12557#
EMS210	077564	12557#
EMS211	077575	12557#
EMS212	077613	12557#
EMS213	077636	12557#
EMS214	077661	12557#
EMS215	077677	12557#
EMS216	077720	12557#
EMS217	077750	12557#
EMS22	072676	12557#
EMS220	100003	12557#
EMS221	100023	12557#
EMS222	100073	12557#
EMS223	100124	12557#
EMS224	100201	12557#
EMS23	072712	12557#
EMS24	072740	12557#
EMS25	072767	12557#
EMS26	073004	12557#
EMS27	073025	12557#
EMS3	072110	12557#
EMS30	073036	12557#
EMS31	073046	12557#
EMS32	073075	12557#
EMS33	073124	12557#
EMS34	073152	12557#
EMS35	073223	12557#
EMS36	073252	12557#
EMS37	073301	12557#
EMS4	072153	12557#
EMS40	073327	12557#
EMS41	073356	12557#
EMS42	073404	12557#
EMS43	073433	12557#

EMS44	073462	12557#		
EMS45	073535	12557#		
EMS46	073600	12557#		
EMS47	073627	12557#		
EMS5	072216	12557#		
EMS50	073656	12557#		
EMS51	073705	12557#		
EMS52	073733	12557#		
EMS53	073745	12557#		
EMS54	073757	12557#		
EMS55	074001	12557#		
EMS56	074030	12557#		
EMS57	074105	12557#		
EMS6	072246	12557#		
EMS60	074133	12557#		
EMS61	074146	12557#		
EMS62	074175	12557#		
EMS63	074213	12557#		
EMS64	074241	12557#		
EMS65	074257	12557#		
EMS66	074325	12557#		
EMS67	074375	12557#		
EMS7	072313	12557#		
EMS70	074422	12557#		
EMS71	074434	12557#		
EMS72	074447	12557#		
EMS73	074457	12557#		
EMS74	074474	12557#		
EMS75	074513	12557#		
EMS76	074521	12557#		
EMS77	074552	12557#		
EMTVEC=	000030	937#	3242*	3243*
EMT1	065430	1521	12557#	
EMT10	065500	1570	12557#	
EMT100	066476	1967	12557#	
EMT101	066514	1974	12557#	
EMT102	066532	1981	12557#	
EMT103	066542	1988	12557#	
EMT104	066554	1995	12557#	
EMT105	066572	2002	12557#	
EMT106	066602	2009	12557#	
EMT107	066612	2016	12557#	
EMT11	065504	1577	12557#	
EMT110	066632	2023	12557#	
EMT111	066644	2030	12557#	
EMT112	066652	2037	12557#	
EMT113	066660	2044	12557#	
EMT114	066674	2051	12557#	
EMT115	066702	2058	12557#	
EMT116	066712	2065	12557#	
EMT117	066722	2072	12557#	
EMT12	065510	1584	12557#	
EMT120	066732	2079	12557#	
EMT121	066742	2086	12557#	
EMT122	066752	2093	12557#	
EMT123	066762	2100	12557#	

```
EMT124  066772            2107   12557#
EMT125  067002            2114   12557#
EMT126  067012            2121   12557#
EMT127  067024            2128   12557#
EMT13   065516            1591   12557#
EMT130  067036            2135   12557#
EMT131  067050            2142   12557#
EMT132  067062            2149   12557#
EMT133  067074            2156   12557#
EMT134  067106            2163   12557#
EMT135  067120            2170   12557#
EMT136  067132            2177   12557#
EMT137  067144            2184   12557#
EMT14   065530            1598   12557#
EMT140  067154            2191   12557#
EMT141  067164            2198   12557#
EMT142  067174            2205   12557#
EMT143  067204            2212   12557#
EMT144  067214            2219   12557#
EMT145  067224            2226   12557#
EMT146  067234            2233   12557#
EMT147  067244            2240   12557#
EMT15   065536            1605   12557#
EMT150  067254            2247   12557#
EMT151  067264            2254   12557#
EMT152  067276            2261   12557#
EMT153  067310            2268   12557#
EMT154  067326            2275   12557#
EMT155  067344            2282   12557#
EMT156  067356            2289   12557#
EMT157  067370            2296   12557#
EMT16   065544            1612   12557#
EMT160  067402            2303   12557#
EMT161  067412            2310   12557#
EMT162  067424            2317   12557#
EMT163  067436           12557#
EMT164  067446            2332   12557#
EMT165  067454            2339   12557#
EMT166  067462            2346   12557#
EMT167  067470            2353   12557#
EMT17   065554            1619   12557#
EMT170  067476           12557#
EMT171  067506            2367   12557#
EMT172  067516            2374   12557#
EMT173  067526            2381   12557#
EMT174  067536            2388   12557#
EMT175  067550            2395   12557#
EMT176  067556            2402   12557#
EMT177  067564            2409   12557#
EMT2    065434            1528   12557#
EMT20   065564            1626   12557#
EMT200  067576            2416   12557#
EMT201  067610            2423   12557#
EMT202  067622            2430   12557#
EMT203  067634            2437   12557#
EMT204  067646            2444   12557#
```

EMT205	067664	2451	12557#
EMT206	067674	2458	12557#
EMT207	067704	2465	12557#
EMT21	065574	1633	12557#
EMT210	067716	2472	12557#
EMT211	067724	2479	12557#
EMT212	067740	2486	12557#
EMT213	067754	2493	12557#
EMT214	067764	2500	12557#
EMT215	070004	2507	12557#
EMT216	070016	2514	12557#
EMT217	070026	2521	12557#
EMT22	065604	1640	12557#
EMT220	070036	2528	12557#
EMT221	070046	2535	12557#
EMT222	070056	2542	12557#
EMT223	070066	2549	12557#
EMT224	070076	12557#	
EMT225	070100	12557#	
EMT226	070102	12557#	
EMT227	070104	12557#	
EMT23	065614	1647	12557#
EMT230	070106	12557#	
EMT231	070110	12557#	
EMT232	070112	12557#	
EMT233	070114	12557#	
EMT234	070116	12557#	
EMT235	070120	12557#	
EMT236	070122	12557#	
EMT237	070124	12557#	
EMT24	065624	1654	12557#
EMT240	070126	12557#	
EMT241	070130	12557#	
EMT242	070132	12557#	
EMT243	070134	12557#	
EMT244	070136	12557#	
EMT245	070140	12557#	
EMT246	070142	2682	12557#
EMT247	070152	2689	12557#
EMT25	065634	1661	12557#
EMT250	070164	2696	12557#
EMT251	070200	2703	12557#
EMT252	070206	2710	12557#
EMT253	070214	2717	12557#
EMT254	070232	2724	12557#
EMT255	070242	2731	12557#
EMT256	070252	2738	12557#
EMT257	070262	2745	12557#
EMT26	065644	1668	12557#
EMT260	070274	2752	12557#
EMT261	070306	2759	12557#
EMT262	070326	2766	12557#
EMT263	070336	2773	12557#
EMT264	070346	2780	12557#
EMT265	070366	2787	12557#
EMT266	070406	2794	12557#

EMT267	070420	2802	12557#	
EMT27	065654	1675	12557#	
EMT270	070436	2809	12557#	
EMT271	070452	2817	12557#	
EMT272	070470	2824	12557#	
EMT273	070506	2831	12557#	
EMT274	070524	2839	12557#	
EMT275	070542	2847	12557#	
EMT276	070554	2855	12557#	
EMT277	070570	2864	12557#	
EMT3	065442	1535	12557#	
EMT30	065664	1682	12557#	
EMT300	070606	2872	12557#	
EMT301	070626	2880	12557#	
EMT302	070650	2887	12557#	
EMT303	070662	2894	12557#	
EMT304	070674	2901	12557#	
EMT305	070706	2908	12557#	
EMT306	070720	2915	12557#	
EMT307	070730	2922	12557#	
EMT31	065674	1689	12557#	
EMT310	070750	2929	12557#	
EMT311	070762	2936	12557#	
EMT312	070774	2943	12557#	
EMT313	071006	2950	12557#	
EMT314	071026	2957	12557#	
EMT315	071040	2964	12557#	
EMT316	071052	2971	12557#	
EMT317	071064	2978	12557#	
EMT32	065704	1696	12557#	
EMT320	071074	2985	12557#	
EMT321	071104	2992	12557#	
EMT322	071114	2999	12557#	
EMT323	071122	3006	12557#	
EMT324	071132	3013	12557#	
EMT325	071144	3020	12557#	
EMT326	071156	3027	12557#	
EMT327	071174	3034	12557#	
EMT33	065714	1703	12557#	
EMT330	071206	3041	12557#	
EMT331	071216	3048	12557#	
EMT332	071230	3055	12557#	
EMT333	071242	3062	12557#	
EMT334	071260	3069	12557#	
EMT335	071276	3076	12557#	
EMT336	071316	2325	3083	12557#
EMT337	071326	3090	12557#	
EMT34	065724	1710	12557#	
EMT340	071336	3097	12557#	
EMT341	071350	3104	12557#	
EMT342	071356	3111	12557#	
EMT343	071370	3118	12557#	
EMT344	071402	3125	12557#	
EMT345	071414	3132	12557#	
EMT346	071424	3139	12557#	
EMT347	071440	3146	12557#	

```
EMT35   065734            1717   12557#
EMT350  071452            3153   12557#
EMT351  071470            3160   12557#
EMT352  071500            3167   12557#
EMT353  071504            3174   12557#
EMT354  071514            3181   12557#
EMT36   065744            1724   12557#
EMT37   065754            1731   12557#
EMT4    065450            1542   12557#
EMT40   065764            1738   12557#
EMT41   065772            1745   12557#
EMT42   066002            1752   12557#
EMT43   066014            1759   12557#
EMT44   066024            1766   12557#
EMT45   066034            1773   12557#
EMT46   066044            1780   12557#
EMT47   066054            1787   12557#
EMT5    065456            1549   12557#
EMT50   066062            1794   12557#
EMT51   066072            1801   12557#
EMT52   066104            1808   12557#
EMT53   066122            1816   12557#
EMT54   066140            1823   12557#
EMT55   066150            1830   12557#
EMT56   066162            1837   12557#
EMT57   066200            1844   12557#
EMT6    065464            1556   12557#
EMT60   066210            1852   12557#
EMT61   066226            1860   12557#
EMT62   066246            1868   12557#
EMT63   066262           12557#
EMT64   066264            1882   12557#
EMT65   066304            1889   12557#
EMT66   066324            1896   12557#
EMT67   066336            1903   12557#
EMT7    065472            1563   12557#
EMT70   066350            1910   12557#
EMT71   066360            1917   12557#
EMT72   066370            1924   12557#
EMT73   066406            1932   12557#
EMT74   066424            1939   12557#
EMT75   066434            1946   12557#
EMT76   066454            1953   12557#
EMT77   066464            1960   12557#
ENRGDT  065430           12557#
EQUALS  063571           12557#
ERR   = 040000            1012#  9272   9274   9290   9324
ERRNMB  033734            7831*  7832*  7835   7847   7852   7944#
ERRTYP  033176            7810#  12086
ERRVEC= 000004             930#  3256   3257*  3268*  3536   3537   3538*  3539*  3550*  3551*  3555*  3556*  3582
                          3583   3584*  3585*  3589*  3590*  3601*  3602*  8844   8845   8849*  8850*  8881*  8882*
                          8917   8918   8922*  8923*  8948*  8949*  8960   8961   8962*  8963*  8969*  8977*  8978*
                          8992   8993   8994*  8995*  9019*  9020*  9867   9868   9871*  9872*  9905*  9906*  10153
                         10154  10155* 10156* 10167* 10168* 11975  11976* 11978* 11981*
ERTY00  033742            7817   7949#
ERTY01  033750            7825   7950#
```

```
ERTY02  033760          7834    7952#
ERTY03  033767          7840    7954#
ERTY04  033775          7936    7956#
ESRC   = 004000         1086#
FER    = 000020         1036#   1042    7145    7155    7283    7293    7434    7444    8369    9604    11202   11219   11222
                        11245   11248
FIND   = 000001         3672#   3674#   3709#   3710    3741#   3742    3748#   3749    3755#   3756    3784#   3785    3791#
                        3792    3798#   3799    3807#   3810    3840#   3842#   3877#   3878    3909#   3910    3916#   3917
                        3923#   3924    3952#   3953    3959#   3960    3966#   3967    3975#   3978    4008#   4010#   4045#
                        4046    4078#   4079    4085#   4086    4092#   4093    4122#   4123    4129#   4130    4136#   4137
                        4145#   4148    4178#   4180#   4215#   4216    4247#   4248    4254#   4255    4261#   4262    4289#
                        4290    4296#   4297    4303#   4304    4338#   4340#   4375#   4376    4407#   4408    4414#   4415
                        4421#   4422    4452#   4453    4463#   4464    4526#   4528#   4563#   4564    4595#   4596    4602#
                        4603    4609#   4610    4638#   4639    4645#   4646    4652#   4653    4661#   4664    4694#   4696#
                        4731#   4732    4763#   4764    4770#   4771    4777#   4778    4805#   4806    4812#   4813    4819#
                        4820    4854#   4856#   4891#   4892    4923#   4924    4930#   4931    4937#   4938    4968#   4969
                        4979#   4980    5042#   5044#   5079#   5080    5111#   5112    5118#   5119    5125#   5126    5153#
                        5154    5160#   5161    5167#   5168    5201#   5203#   5238#   5239    5270#   5271    5277#   5278
                        5284#   5285    5312#   5313    5319#   5320    5326#   5327    5360#   5362#   5397#   5398    5429#
                        5430    5436#   5437    5443#   5444    5471#   5472    5478#   5479    5485#   5486    5519#   5521#
                        5558#   5559    5591#   5592    5598#   5599    5605#   5606    5633#   5634    5640#   5641    5647#
                        5648    5684#   5686#   5722#   5723    5729#   5730    5736#   5737    5765#   5766    5772#   5773
                        5779#   5780    5788#   5791    5829#   5831#   5867#   5868    5874#   5875    5881#   5882    5910#
                        5911    5917#   5918    5924#   5925    5933#   5936    5974#   5976#   6012#   6013    6019#   6020
                        6026#   6027    6055#   6056    6062#   6063    6069#   6070    6078#   6081    6107#   6109#   6155#
                        6156    6162#   6163    6169#   6170    6191#   6193#   6237#   6238    6244#   6245    6251#   6252
                        6286#   6288#   6324#   6325    6331#   6332    6338#   6339    6347#   6349#   6378#   6379    6385#
                        6386    6392#   6393    6433#   6435#   6471#   6472    6478#   6479    6485#   6486    6494#   6496#
                        6525#   6526    6532#   6533    6539#   6540    6580#   6582#   6618#   6619    6625#   6626    6632#
                        6633    6641#   6643#   6672#   6673    6679#   6680    6686#   6687    6726#   6728#   6776#   6777
                        6783#   6784    6790#   6791    6838#   6839    6845#   6846    6852#   6853    6861#   6864    6901#
                        6903#   6952#   6953    6966#   6967    6975#   6977#   7006#   7007    7020#   7021    7054#   7056#
                        7092#   7093    7099#   7100    7106#   7107    7136#   7137    7147#   7148    7158#   7159    7192#
                        7194#   7230#   7231    7237#   7238    7244#   7245    7274#   7275    7285#   7286    7296#   7297
                        7331#   7333#   7378#   7379    7385#   7386    7392#   7393    7421#   7422    7436#   7437    7451#
                        7452    7471#   7472    7489#   7490    7516#   7518#   7567#   7569#   7614#   7615    7621#   7622
                        7628#   7629    7657#   7658    7668#   7669    7686#   7687    7711#   7713#
FMT16  = 010000         1111#   3830    3998    4168    4328    4516    4684    4844    5032    5191    5350    5509    5674
                        5819    5964    6118    6202    6423    6570    6716    6891    7116    7182    7321    7430    7510
                        7557    7706    8205    8634    8636    8707    9964    11040
FNCDTB  064672          9318    12557#
FNCMSK= 000077           954#   10751   11195
F0     = 000002          952#   9217
F1     = 000004          951#   9217
F2     = 000010          950#   9217    9791
F3     = 000020          949#   9217
F4     = 000040          948#   9217
GENBUF  036620          3668    3836    4004    4174    4334    4522    4690    4850    5038    5197    5356    5515    5680
                        5825    5970    6282    6429    6576    6722    6897    7050    7188    7327    7514    7563    7710
                        8618#
GET     037516          3629    3701    3733    3776    3869    3901    3944    4037    4070    4114    4207    4239    4281
                        4367    4399    4444    4484    4555    4587    4630    4723    4755    4797    4883    4915    4960
                        4999    5071    5103    5145    5230    5262    5304    5389    5421    5463    5550    5583    5625
                        5714    5757    5859    5902    6004    6047    6147    6229    6316    6370    6463    6517    6610
                        6664    6768    6830    6944    6998    7084    7128    7222    7266    7370    7413    7606    7649
                        8028    8052    8079    8102    8128    8232    8265    8301    8335    8835#
```

```
GETBUF  001330          1423#  8847
GETINX  001506          1494#  3626*  3627*  4482*  4483*  4997*  4998*  8050*  8051*  8100*  8101*  8793  8848
GETSTS  037432          3696   3864   4032   4202   4362   4550   4718   4878   5066   5225   5384   5545  5710
                        5855   6000   6144   6225   6312   6366   6459   6513   6606   6660   6764   6940  6994
                        7080   7218   7366   7602   8027   8078   8121   8219   8789#
GNS     = ****** U      1261   3294   7769   7776   12441  12442  12443  12444  12445  12446  12448  12450 12451
                       12452  12453  12454  12455
GO      = 000001         953#  3681   3718   3765   3849   3886   3933   4017   4055   4103   4187   4224  4271
                        4347   4384   4434   4535   4572   4619   4703   4740   4787   4863   4900   4950  5051
                        5088   5135   5210   5247   5294   5369   5406   5453   5530   5568   5615   5693  5746
                        5838   5891   5983   6036   6121   6205   6295   6356   6442   6503   6589   6650  6719
                        6735   6750   6798   6814   6921   6984   7063   7117   7201   7255   7349   7402  7522
                        7585   7638   7717   8062   8111   8223   8256   8290   8324   9005   9116   9128  9133
                        9149   9239   9242   11196
GTSWR   = 104407        3289   3510  12448#
HCE     = 000200        1033#  1042   7469   7479   7666   7676   8369   9567   9572   9587   9589   9602  9617
                       11202  11232  11235  11245  11248  12557
HCI     = 002000        1113#  9565  11198
HCRC    = 000400        1032#  1042   8369   9574  11202  11206  11209  11245  11248
HELP      103564        3371  12557#
HELPQS    063601        3365  12557#
HT      = 000011         840#  7873  11912  11953
IAE     = 002000        1030#  9471   9768   9783   9809   9955   9972  10471  10525  10528  11029  11044 12557
IDXMSK= 000077          1189#  8861   8864   8934   8937
IE      = 000100        1203#  1236#
ILF     = 000001        1040#  9343   9345  10365  10425  10429  10471  10492  10495  10985  11002  11005 12557
ILF02   = 000002         959#
ILF24   = 000024         969#
ILF26   = 000026         970#
ILF30   = 000030         972#
ILF32   = 000032         973#
ILF34   = 000034         974#
ILF36   = 000036         975#
ILF40   = 000040         976#
ILF42   = 000042         977#
ILF44   = 000044         978#
ILF46   = 000046         979#
ILF54   = 000054         982#
ILF56   = 000056         983#
ILF64   = 000064         986#
ILF66   = 000066         987#
ILF74   = 000074         990#
ILF76   = 000076         991#
ILR     = 000002        1039#  10365  10411  10415  10679  10683  10686  10985  10988  10991
ILRG50= 000050          1175#
ILRG52= 000052          1176#
ILRG54= 000054          1177#
ILRG56= 000056          1178#
ILRG60= 000060          1179#
ILRG62= 000062          1180#
ILRG64= 000064          1181#
ILRG66= 000066          1182#
ILRG70= 000070          1183#
ILRG72= 000072          1184#
ILRG74= 000074          1185#
ILRG76= 000076          1186#
```

```
IOTVEC= 000020   935#  3240*  3241*
IPCK0 = 000001  1240#
IPCK1 = 000002  1239#
IPCK2 = 000004  1238#
IPCK3 = 000010  1237#
IR    = 000100  1217#  10228
IVC   = 010000  1146#  6959   6963   7013   7017   9386   9387   9990   10113  10539  10546  10549  10618
                10968  10972  11400  11481  12557
LBC   = 002000  1148#  11086  11104  11109
LBT   = 002000  1016#  9546   9770   9772
LF    = 000012   841#  7871   11656  11947  11953  12557
LS    = 000004  1095#  10252  10804
LSC   = 004000  1147#  11086  11120  11123
LST   = 000002  1096#  10252  10804
MCLK  = 004000  1068#
MCPE  = 020000  1197#  9116   9133   9146   9149   10883  10886
MDF   = 000100  1073#
MDPE  = 000400  1214#  9642   11292  11295
MEDENB  001474  1482#  3522*  8061*  8184   8440*
MFGFIL  101544  8206   8409   8422*  8429*  8490   12557#
MI    = 000004  1076#
MIXED   065002  12557#
MOC   = 000400  1071#
MOH   = 020000  1103#
MOL   = 010000  1014#  8361   10051  10061  10062  10067  10072  10100  10351  10354  10515  10592  10593
                10600  10605  10764  10945  11106  11359  11360  11382  11387  11458  11459  11463  11468
MRD   = 002000  1069#
MS    = 000040  1074#
MSC   = 000002  1077#
MSE   = 100000  1154#  7428   8409   8632
MSER  = 000200  1072#
MUR   = 001000  1070#
MWD   = 000010  1094#  10253  10805
MWP   = 000010  1075#
MXF   = 001000  1213#  9531   10906  10920  10923
NDTMSK= 115760  1042#
NED   = 010000  1210#  3321   3591   9098   9102   9880
NEM   = 004000  1211#  9516
NOP   = 000000   958#
NOTEX   064645  3329   12557#
NSA   = 100000  1101#
OCC   = 100000  1082#
OFD   = 000200  1114#  6869   6871
OFFSET= 000014   964#  6735   6798
OM    = 000001  1021#  10592  10648  10651  10763  11307  11310
ONES    065042  4520   4688   4848   12557#
OPE   = 020000  1145#
OPI   = 020000  1027#  9370   10045  10069  10365  10383  10387  10471  10508  10511  10602  10940  10943
                11384  11465  12557
OR    = 000200  1216#
PACACK= 000022   968#  8256
PAKACK= 000022   967#   968   8062
PAR   = 000010  1037#  8372   9161   9165   9177   9937   10471  10476  10481  10897  10902
PAT   = 000020  1219#
PDA   = 000400  1089#
PGE   = 002000  1212#
```

```
PGM    = 001000          1017#  10763
PHA    = 000200          1090#
PIP    = 020000          1013#  8109   8287   10061  10082  10087  10592  10663  10668  10763  11359  11365  11370
                         11458  11495  11500
PIRQ   = 177772           847#
PIRQVE= 000240            941#
PLFS   = 002000          1087#
PRIERR  040512           3741   3784   3909   3952   4078   4122   4247   4289   4407   4452   4595   4638   4763
                         4805   4923   4968   5111   5153   5270   5312   5429   5471   5591   5633   5722   5765
                         5867   5910   6012   6055   6155   6237   6324   6378   6471   6525   6618   6672   6776
                         6838   6952   7006   7092   7136   7230   7274   7378   7421   7614   7657   9064#
PROMPT  063573           3446   3455   12557#
PR0    = 000000           864#
PR1    = 000040           865#
PR2    = 000100           866#
PR3    = 000140           867#  11631
PR4    = 000200           868#
PR5    = 000240           869#
PR6    = 000300           870#  3227   3517   3539   3585   8850   8923   8963   8995   9872   10156  11637
PR7    = 000340           871#
PS     = 177776           844#   845
PSEL   = 002000          1199#
PSW    = 177776           845#
PUT     037766           3688   3724   3767   3856   3892   3935   4024   4061   4105   4194   4230   4272   4354
                         4390   4435   4542   4578   4621   4710   4746   4788   4870   4906   4951   5058   5094
                         5136   5217   5253   5295   5376   5412   5454   5537   5574   5616   5702   5748   5847
                         5893   5992   6038   6136   6217   6304   6358   6451   6505   6598   6652   6742   6756
                         6805   6821   6913   6932   6986   7072   7119   7210   7257   7358   7404   7531   7594
                         7640   7726   8065   8114   8224   8257   8291   8325   8911#
PUTBUF  001400           1450#  8195   8444*  8920
PUTINX  001535           1499#  3682   3719   3850   3887   4018   4056   4188   4225   4348   4385   4536   4573
                         4704   4741   4864   4901   5052   5089   5211   5248   5370   5407   5531   5569   5694
                         5839   5984   6123   6206   6296   6443   6590   6736   6751   6799   6816   6911*  6912*
                         6923   7064   7202   7350   7523   7586   7718   8063*  8064*  8112*  8113*  8208   8921
                         9168
PWRVEC= 000024            936#  3246*  3247*  12460*  12461*  12470*  12476*  12488*  12489*
QSTMRK  063577           3360   3465   12557#
RCLSTS  050452           8135   8310   10462#
RD     = 000070           988#
RDCHR  = 104411          3352   3366   3376   3379   3447   3456   12313  12451#
RDLIN  = 104412          12383  12452#
RDOCT  = 104413          3390   3406   3427   12453#
RDY    = 000200          1202#  9005   9006   9112   9115   10203
READY   007124           3357   3514#  7744   11649
RECAL  = 000006           961#  8111   8290
RESREG= 104415           7940   12455#
RESVEC= 000010            931#
REX    = 010000          1085#
RG     = 040000          1083#
RGDTPT  065002           12557#
RH     = 000072           989#  3765   3933   4103   4619   5746   5891   6036   6121   6205   6356   6503   6650
                         6814   6984   7117   7255   7402   7638   8324
RIP    = 000020           966#
RLEASE= 000012            963#
RMAS   = 000016          1163#
RMAS1   001346           1433#  10784  12557
```

```
RMASO   001416  1460#
RMBA  = 000004  1245#  3544*  3545   3721   3889   4058   4227   4387   4575   4743   4903   5091   5250
                5409   5571   5699   5844   5989   6127   6211   6301   6448   6595   6753   6818   6929
                7069   7207   7355   7528   7591   7722   8213
RMBAE = 000050  1248#
RMBAI   001334  1428#  4478   4993   9718   9720   10214  12557
RMBAO   001404  1455#  3664*  3766*  3832*  3934*  4000*  4104*  4170*  4330*  4518*  4620*  4686*  4846*
                5034*  5193*  5352*  5511*  5676*  5747*  5821*  5892*  5966*  6037*  6120*  6204*  6278*
                6357*  6425*  6504*  6572*  6651*  6718*  6815*  6872*  6893*  6985*  7046*  7118*  7184*
                7256*  7323*  7403*  7512*  7559*  7639*  7708*  8206*  8624   8711   9714   9717
RMCC  = 000036  1170#
RMCCI   001366  1441#
RMCCO   001436  1468#
RMCS1 = 000000  1159#  1243#  3323   3541*  3587   3686   3722   3854   3890   4022   4059   4192   4228
                4352   4388   4540   4576   4708   4744   4868   4904   5056   5092   5215   5251   5374
                5410   5535   5572   5700   5845   5990   6129   6212   6302   6449   6596   6740   6754
                6803   6819   6930   7070   7208   7356   7529   7592   7724   8063   8112   8214   8852
                8925   9004   9878
RMCS1I  001330  1426#  9091   9093   9095   9112   9114   9117   9128   9132   9134   9146   9148   9150
                9239   9241   9252   9295   9701   10201  10750  10883  10885  10887  10906  12557
RMCS1O  001400  1453#  3665*  3681*  3718*  3765*  3833*  3849*  3886*  3933*  4001*  4017*  4055*  4103*
                4171*  4187*  4224*  4271*  4331*  4347*  4384*  4434*  4519*  4535*  4572*  4619*  4687*
                4703*  4740*  4787*  4847*  4863*  4900*  4950*  5035*  5051*  5088*  5135*  5194*  5210*
                5247*  5294*  5353*  5369*  5406*  5453*  5512*  5530*  5568*  5615*  5677*  5693*  5746*
                5822*  5838*  5891*  5967*  5983*  6036*  6121*  6205*  6279*  6295*  6356*  6426*  6442*
                6503*  6573*  6589*  6650*  6719*  6735*  6750*  6798*  6814*  6894*  6921*  6984*  7047*
                7063*  7117*  7185*  7201*  7255*  7324*  7349*  7402*  7513*  7522*  7560*  7585*  7638*
                7709*  7717*  8062*  8111*  8223*  8256*  8290*  8324*  8470   8628   9216   9732   11174
                11194
RMCS2 = 000010  1246#  3318*  3319*  3320   3546*  3547   3588   8851   8924   9877*  9879   10158*  10160*
RMCS2I  001340  1430#  4461   4469   4471   4977   4985   4987   9073   9098   9100   9101   9287   9358
                9459   9502   9517   9532   9643   9715   10223  10920  10922  10924  11292  11294  11296
                11346  11348  11350  12557
RMCS2O  001410  1457#
RMCS3 = 000052  1249#
RMDA  = 000006  1160#  3683   3851   4019   4189   4349   4537   4705   4865   5053   5212   5371   5532
                5695   5840   5985   6124   6207   6297   6444   6591   6737   6800   6925   7065   7203
                7351   7524   7587   7719   8209
RMDAI   001336  1429#  9814   12557
RMDAO   001406  1456#  3661*  3829*  3997*  4167*  4327*  4515*  4683*  4843*  5031*  5190*  5349*  5508*
                5673*  5678   5798*  5799   5818*  5823   5943*  5944   5963*  6117*  6201*  6275*  6280
                6402*  6403   6422*  6427   6549*  6550   6569*  6715*  6720   6890*  6895   7043*  7181*
                7320*  7556*  8202*  8383*  8384   8386   8404   8412*  8464   8591*  8627   9740   9741
                9960   9962   9966   11034  11036  11038
RMDB  = 000022  1247#  3548*  3549   4482   4997   8800
RMDBI   001352  1435#  4489   5004
RMDBO   001422  1462#
RMDC  = 000034  1169#  3684   3852   4020   4190   4350   4538   4706   4866   5054   5213   5372   5533
                5696   5841   5986   6125   6208   6298   6445   6592   6738   6801   6926   7066   7204
                7352   7525   7588   7720   8210
RMDCI   001364  1440#  9826   12557
RMDCO   001434  1467#  3660*  3828*  3996*  4166*  4326*  4514*  4682*  4842*  5030*  5189*  5348*  5507*
                5528   5529*  5567*  5672*  5817*  5962*  5968   6088*  6089   6116*  6200*  6274*  6421*
                6568*  6574   6695*  6696   6714*  6889*  7042*  7180*  7319*  7555*  8203*  8463   8590*
                8626   9742   9957   11031
RMDS  = 000012  1161#  8050   8100
```

```
RMDSI   001342          1431#  8059   8109   8254   8287   8361   9130   9226   9228   9273   9290   9324   9328
                        9384   9411   9546   9771   9996  10051  10060  10067  10071  10073  10082  10086  10088
                       10097  10099  10101  10111  10115  10117  10296  10337  10339  10341  10351  10353  10355
                       10515  10553  10591  10600  10604  10606  10616  10620  10621  10632  10634  10636  10648
                       10650  10652  10663  10667  10669  10762  10945  10947  10976  11106  11172  11307  11309
                       11311  11358  11365  11369  11371  11382  11386  11388  11398  11402  11404  11457  11463
                       11467  11469  11479  11483  11484* 11485  11495  11499  11501  12557
RMDSO   001412          1458#
RMDT  = 000026          1166#  3626
RMDTI   001356          1437#  3628*  3633   3635   3637   3639   12557
RMDTO   001426          1464#
RMEC1 = 000044          1173#
RMEC1I  001374          1444#  8712   12557
RMEC1O  001444          1471#
RMEC2 = 000046          1174#  8795
RMEC2I  001376          1445#  8733   8794   10264  10828  12557
RMEC2O  001446          1472#
RMER1 = 000014          1162#  9169
RMER1I  001344          1432#  7145   7153   7154   7283   7291   7292   7434   7442   7443   7469   7477   7478
                        7666   7674   7675   8368   8701   8703   9161   9176   9178   9268   9344   9371   9414
                        9429   9472   9549   9575   9590   9605   9658   9677   9680   9768   9783   9798   9809
                        9937   9944   9971  10045  10048  10069  10241  10365  10369  10371  10372  10383  10385
                       10386  10397  10399  10400  10411  10413  10414  10425  10427  10428  10471  10476  10480
                       10482  10492  10494  10496  10508  10510  10512  10525  10527  10529  10602  10679  10683
                       10685  10687  10697  10699  10701  10711  10715  10717  10775  10897  10901  10903  10940
                       10942  10944  10985  10988  10990  10992  11002  11004  11006  11016  11018  11020  11043
                       11133  11136  11140  11142  11151  11153  11155  11165  11167  11168  11202  11206  11208
                       11210  11219  11221  11223  11232  11234  11236  11245  11247  11249  11267  11269  11271
                       11277  11333  11335  11337  11384  11465  11576  12557
RMER1O  001414          1459#  9165
RMER2 = 000042          1172#
RMER2I  001372          1443#  6959   6961   6962   7013   7015   7016   7449   7458   7459   8370   8375   9163
                        9270   9388   9444   9620   9939   9990   9993  10004  10032  10035  10084  10113  10286
                       10478  10539  10546  10548  10550  10563  10565  10567  10577  10579  10581  10618  10665
                       10713  10837  10899  10968  10971  10973  11057  11061  11086  11090  11092  11094  11104
                       11108  11110  11120  11122  11124  11138  11320  11322  11324  11367  11400  11481  11497
                       11511  11517  11519  11521  11531  11533  11535  11579  12557
RMER2O  001442          1470#
RMLA  = 000020          1164#
RMLAI   001350          1434#  12557
RMLAO   001420          1461#
RMMR1 = 000024          1165#  6911   6924
RMMR1I  001354          1436#  10251  10803  12557
RMMR1O  001424          1463#  6910*  6922*
RMMR2 = 000040          1171#
RMMR2I  001370          1442#  10273  10815  12557
RMMR2O  001440          1469#
RMOF  = 000032          1168#  3685   3853   4021   4191   4351   4539   4707   4867   5055   5214   5373   5534
                        5697   5842   5987   6126   6209   6299   6446   6593   6739   6802   6927   7067   7205
                        7353   7526   7589   7721   8212
RMOFI   001362          1439#  8705   8707   9565   9675  11198  11274  12557
RMOFO   001432          1466#  3662*  3830*  3998*  4168*  4328*  4516*  4684*  4844*  5032*  5191*  5350*  5509*
                        5674*  5819*  5964*  6118*  6202*  6276*  6423*  6570*  6716*  6869   6871*  6891*  7044*
                        7116*  7182*  7254*  7321*  7510*  7557*  7706*  8205*  8634   9964  11040  12557
RMR   = 000004          1038#  10365  10397  10401  10679  10697  10700  10985  11016  11019
RMSN  = 000030          1167#
```

```
RMSN1   001360          1438#  12557
RMSNO   001430          1465#
RMWC  = 000002          1244#  3542*  3543   3720   3888   4057   4226   4386   4574   4742   4902   5090   5249
                        5408   5570   5698   5843   5988   6128   6210   6300   6447   6594   6752   6817   6928
                        7068   7206   7354   7527   7590   7723   8211
RMWC1   001332          1427#  8750   9699   9711   9729   9793   12557
RMWCO   001402          1454#  3663*  3831*  3999*  4054*  4102*  4169*  4329*  4517*  4685*  4845*  5033*  5192*
                        5351*  5510*  5675*  5820*  5965*  6119*  6203*  6277*  6424*  6571*  6717*  6892*  7045*
                        7183*  7322*  7511*  7558*  7707*  8204*  8466   8625   8751   9712   9730
RQA   = 100000          1123#  10274  10816
RQB   = 040000          1124#  10274  10816
RTC   = 000016           965#
SADMSK= 000037          1007#
SAVREG= 104414          7811   12454#
SA1   = 000001          1002#
SA16  = 000020           998#
SA2   = 000002          1001#
SA4   = 000004          1000#
SA8   = 000010           999#
SC    = 100000          1195#  9116   9133   9149   10202
SCTMSG  063460         12557#
SCTMSK= 003700          1058#
SC0   = 000100          1056#
SC1   = 000200          1055#
SC2   = 000400          1054#
SC3   = 001000          1053#
SC4   = 002000          1052#
SEARCH= 000030           971#  9292   9693
SECERR  041344          3755   3798   3923   3966   4092   4136   4261   4303   4421   4609   4652   4777   4819
                        4937   5125   5167   5284   5326   5443   5485   5605   5647   5736   5779   5881   5924
                        6026   6069   6169   6251   6338   6392   6485   6539   6632   6686   6790   6852   6966
                        7020   7106   7158   7244   7296   7392   7489   7628   7686   9212#
SEEK  = 000004           960#  3681   3849   4017   4187   4347   4535   4703   4863   5051   5210   5369   5530
SEKSTS  045420          3709   3877   4045   4215   4375   4563   4731   4891   5079   5238   5397   5558   9926#
SHUT    057372          7792   11646#
SHUT2   057364         11645#  12154  12237
SIZCLK  040204          3497   8959#
SKI   = 040000          1144#  9956   9959   9975   10005  10007  10011  10022  10084  10287  10539  10563  10566
                        10665  11030  11033  11047  11058  11060  11061  11063  11075  11367  11497  11512  11531
                        11534
SNGPRT= 020024          1106#  3633   3637   3641
STACK = 001100           835#  3238   3531   3571   3615   3653   3821   3989   4159   4319   4507   4675   4835
                        5023   5182   5341   5500   5665   5810   5955   6100   6185   6267   6414   6561   6707
                        6882   7035   7173   7311   7547
STANDA  006144          3307   3342#
START   005324          1281   3222#  3560   11655
STCDRV  056416         11450#
STKLMT= 177774           846#
STOP    057334         11628#  11969
STSD1   101434         12557#
STSD2   101452         12557#
STSD3   101472         12557#
STSD4   101504         12557#
STSF    101535         12557#
STSH1   101067         12557#
STSH2   101146         12557#
```

```
STSM3   101237            12557#
STSM4   101277            12557#
SWR     001154             1334#  3236   3258*  3260   3266*  3273*  3287   3508   7812   11970  11984  11987  12002
                          12009  12077  12084  12096  12100  12157  12196  12251* 12468  12481*
SWREG   000176             1263#  3266   3287   3508   12157  12196  12219
SW0   = 000001              899#
SW00  = 000001              889#   899
SW01  = 000002              888#   898
SW02  = 000004              887#   897
SW03  = 000010              886#   896
SW04  = 000020              885#   895
SW05  = 000040              884#   894
SW06  = 000100              883#   893
SW07  = 000200              882#   892
SW08  = 000400              881#   891
SW09  = 001000              880#   890
SW1   = 000002              898#
SW10  = 002000              879#
SW11  = 004000              878#
SW12  = 010000              877#
SW13  = 020000              876#   7812
SW14  = 040000              875#
SW15  = 100000              874#
SW2   = 000004              897#
SW3   = 000010              896#
SW4   = 000020              895#
SW5   = 000040              894#
SW6   = 000100              893#
SW7   = 000200              892#
SW8   = 000400              891#
SW9   = 001000              890#
TADMSK= 003400             1006#
TAG   = 020000             1125#
TAGADR= 001114             1306#  1313
TAP   = 040000             1102#
TA1   = 000400              997#
TA2   = 001000              996#
TA4   = 002000              995#
TBITVE= 000014              932#
TIMOUT  040326             3697   3732   3775   3865   3900   3943   4033   4069   4113   4203   4238   4280   4363
                           4398   4443   4551   4586   4629   4719   4754   4796   4879   4914   4959   5067   5102
                           5144   5226   5261   5303   5385   5420   5462   5546   5582   5624   5713   5756   5858
                           5901   6003   6046   6146   6228   6315   6369   6462   6516   6609   6663   6767   6829
                           6943   6997   7083   7127   7221   7265   7369   7412   7537   7605   7648   7732   8122
                           8300   8334   8988#
TKVEC = 000060              939#   12134* 12135*
TPVEC = 000064              940#
TRAPVE= 000034              938#   3244*  3245*
TRE   = 040000             1196#   9116   9133   9149   9294   9296   9701
TRTVEC= 000014              933#
TST   = 010000             1126#   10275  10817
TSTNMB  033732             7823*  7824*  7826   7943#
TSTPRP  034000             3672   3840   4008   4178   4338   4526   4694   4854   5042   5201   5360   5519   5684
                           5829   5974   6107   6191   6286   6347   6433   6494   6580   6641   6726   6901   6975
                           7054   7192   7331   7516   7567   7711   7991#
TSTQUE  001450             1478#  3480   3481*  3521   3533   3573   3617   3655   3823   3991   4161   4321   4509
```

		4677	4837	5025	5184	5343	5502	5607	5812	5957	6102	6187	6269	6416
		6563	6709	6884	7037	7175	7313	7549	7739	7741*	7745*	9874	10159	10226
		10787												
TST1	007162	3526#	12029											
TST10	012716	4316#	12036											
TST11	013636	4504#	12037											
TST12	014436	4672#	12038											
TST13	015212	4832#	12039											
TST14	016130	5020#	12040											
TST15	016704	5179#	12041											
TST16	017460	5338#	12042											
TST17	020234	5497#	12043											
TST2	007360	3568#	12030											
TST20	021036	5662#	12044											
TST21	021562	5807#	12045											
TST22	022306	5952#	12046											
TST23	023030	6097#	12047											
TST24	023326	6182#	12048											
TST25	023624	6264#	12049											
TST26	024352	6411#	12050											
TST27	025100	6558#	12051											
TST3	007544	3612#	12031											
TST30	025626	6704#	12052											
TST31	026520	6879#	12053											
TST32	027326	7032#	12054											
TST33	030050	7170#	12055											
TST34	030572	7308#	12056											
TST35	031750	7544#	12057											
TST4	007726	3650#	12032											
TST5	010526	3818#	12033											
TST6	011326	3986#	12034											
TST7	012142	4156#	12035											
TYPBN =	104406	7933	12446#											
TYPDS =	104405	7773	7780	7927	12445#									
TYPE =	104401	3281	3301	3329	3351	3354	3360	3365	3368	3371	3375	3378	3386	3389
		3395	3400	3405	3411	3421	3426	3432	3445	3446	3449	3455	3458	3465
		7767	7774	7781	7816	7817	7825	7834	7840	7849	7887	7893	7897	7910
		7916	7918	7936	7938	11650	11722	11788	11857	11917	12079	12087	12151	12163
		12217	12218	12221	12232	12242	12253	12272	12320	12326	12331	12335	12340	12341
		12343	12346	12350	12441#	12490								
TYPOC =	104402	3388	7930	12220	12442#									
TYPON =	104404	12444#												
TYPOS =	104403	3331	3402	3423	7820	7828	7837	7843	12443#					
UBUSQS	063770	3375	12557#											
UNS =	040000	1026#	10242	10365	10369	10373	10679	10711	10716	11133	11136	11141		
UNTMSK=	000007	1227#	9074	9076										
UPE =	020000	1209#	9458											
USE =	040000	1155#	7428	8409	8632									
USRFIL	102554	8423*	8430*	8581	8583	12557#								
U0 =	000001	1224#												
U1 =	000002	1223#												
U2 =	000004	1222#												
VV =	000100	1020#	8059	8254	9384	9996	10061	10062	10111	10116	10337	10340	10553	10592
		10593	10616	10622	10763	10947	10976	11359	11360	11398	11403	11458	11459	11479
		11484												
WC =	000040	1092#	10252	10804										

```
WCD    = 000050          980#
WCE    = 040000         1208#  4461   4470   4977   4986   9357  12557
WCEMI  = 010000         1233#
WCELO  = 004000         1234#
WCF    = 000040         1035#  1042   9428  11333  11336
WCH    = 000052          981#  4271   4434   4787   4950   5135   5294   5453   5615
WD     = 000060          984#
WH     = 000062          985#  3665   3718   3833   3886   4001   4055   4171   4224   4331   4384   4519   4572
                        4687  4740   4847   4900   5035   5088   5194   5247   5353   5406   5512   5568   5677
                        5693  5822   5838   5967   5983   6279   6295   6426   6442   6573   6589   6719   6750
                        6894  6921   7047   7063   7185   7201   7324   7349   7513   7522   7560   7585   7709
                        7717 11196
WLE    = 004000         1029#  1042   9409   9413   9426   9441   9456  11133  11165  11169  11177  12557
WRL    = 004000         1015#  9411  11172
XDPMG    064560         3301  12557#
XSIZ     006014         3311#  3452
ZEROS    065104         3666  3834   4002   4172   4332   5036   5195   5354   5513   7048   7186   7325   7561
                       12557#
$APTHD   001100         1293  1299#
$ASTAT=  ****** U      12533  12548
$ATYC    063236        12504  12506#
$ATY1    063212        12502#
$ATY3    063220        11902  12503#
$ATY4    063230        12092  12505#
$AUTOB   001150         1331#  3291*  3512*  12210  12367
$BASE    001276         1404#  3317   3387   3397*  3532   3572   3616   3654   3822   3990   4160   4320   4508
                        4676  4836   5024   5183   5342   5501   5666   5811   5956   6101   6186   6268   6415
                        6562  6708   6883   7036   7174   7312   7548   8846   8919   8996   9873  10157  12557
$BDADR   001136         1326#  4478*  4479*  4492   4993*  4994*  5007   8761*  12557
$BDDAT   001142         1328#  4471*  4489*  4496   4987*  5004*  5012   6961*  7015*  7153*  7291*  7442*  7458*
                        7477*  7674*  8759*  9073*  9074*  9077   9095*  9101*  9117*  9134*  9150*  9178*  9228*
                        9229*  9241*  9242*  9252*  9253*  9255   9273*  9274*  9275   9295*  9296*  9297   9328*
                        9329*  9330   9344*  9345*  9346   9358*  9359*  9371*  9372*  9388*  9389*  9414*  9415*
                        9429*  9430*  9444*  9445*  9459*  9460*  9472*  9473*  9502*  9503*  9517*  9518*  9532*
                        9533*  9549*  9550*  9575*  9576*  9590*  9591*  9605*  9606*  9620*  9621*  9643*  9644*
                        9658*  9659*  9680*  9681*  9699*  9720*  9771*  9772*  9773   9798*  9799*  9800   9814*
                        9815   9826*  9827   9944*  9971*  9972*  9973   9993*  10004*  10005*  10011  10022  10035*
                       10048*  10073*  10087*  10088*  10101*  10117*  10201*  10202*  10204  10214*  10223*  10231  10241*
                       10242*  10251*  10252*  10254  10264*  10273*  10274*  10276  10286*  10287*  10296*  10297*  10299
                       10341*  10355*  10371*  10385*  10399*  10413*  10427*  10482*  10496*  10512*  10529*  10550*  10567*
                       10581*  10606*  10621*  10636*  10652*  10669*  10687*  10701*  10717*  10750*  10751*  10753  10762*
                       10763*  10765   10775*  10784*  10791*  10794   10803*  10804*  10806   10815*  10816*  10818  10828*
                       10837*  10887*  10903*  10924*  10944*  10973*  10992*  11006*  11020*  11043*  11044*  11045  11057*
                       11058*  11094*  11110*  11124*  11142*  11155*  11167*  11210*  11223*  11236*  11249*  11271*  11296*
                       11311*  11324*  11337*  11350*  11371*  11388*  11404*  11469*  11485*  11501*  11521*  11535*  11591*
                       11606*  12557
$BELL    001212         1351#  12079  12112  12163  12360
$BIN     057620        11718*  11721*  11722  11729#
$CDW1    001302         1406#
$CDW2    001304         1407#
$CHARC   060552        11919*  11929*  11936  11945*  11950#
$CKSWR   061664        12196#  12450
$CMTAG   001114         1314#  3233   3234   3242   3248   3249   3250
$CM3   = 000000         1344#
$CM4   = 000005         1344#  1345#  1346#  1347#  1348#  1349#
$CNTLC   062562        12151  12232  12360#
```

```
$CNTLG  062574          12217  12362#
$CNTLU  062567          12242  12335  12361#
$CPUOP  001250           1378#
$CRLF   001217           1353#   7781   7816   7849   7887   7897   7910   7918   7938  11918  11953  12087  12112
                        12253  12340  12360
$DBLK   060036          11754  11788  11796#
$DDW0   001306           1408#
$DDW1   001310           1409#
$DDW2   001312           1410#
$DDW3   001314           1411#
$DDW4   001316           1412#
$DDW5   001320           1413#
$DDW6   001322           1414#
$DDW7   001324           1415#
$DEVCT  001232           1369#
$DEVM   001300           1405#   3313*  3314*  3326*  3444*  3453*  3471*  3472   3479
$DOAGN  033166           7763    7784   7790#
$DTBL   060026          11757  11792#
$ENDAD  033156           1272    3279   7786#  12107
$ENDCT  033022           3248    7765#
$ENULL  033172           7793#
$ENV    001242           1374#   3285   3306   3506  11897  12089  12511  12535
$ENVM   001243           1375#   3271   3311  11899  11904  12513
$EOP    032766           3561    7755#  11653
$EOPCT  033014           3248*   7762#   7766
$EOSP   032732           3604    3644   7737#
$ERFLG  001117           1317#  11958  11998  12000  12006*  12028  12074*  12112
$ERMAX  001131           1323#   3251*  12000  12023*  12028
$ERROR  061166           3242   12072#
$ERRPC  001132           1324#   7841  12081*  12082*  12083  12112
$ERRTB  001564           1517#   7856
$ERTTL  001126           1321#   7778   7782*  12080*  12112
$ESCAP  001210           1350#   3250*  12022*  12103  12105  12112
$ETABL  001242           1373#
$ETEND  001326           1305    1416#
$FATAL  001224           1366#  12539*
$FFLG   063456          12502*  12505*  12533  12542*  12550#
$FILLC  001172           1342#  11922  11953
$FILLS  001171           1341#  11953
$GDADR  001134           1325#   4477*   4492   4992*   5007   8760*  12557
$GDDAT  001140           1327#   4469*   4470*   4490*   4491*   4496   4985*   4986*   5005*   5006*   5012   6962*   6963*
                         7016*   7017*   7154*   7155*   7292*   7293*   7443*   7444*   7459*   7460*   7478*   7479*   7675*
                         7676*   8758*   9075*   9076*   9077    9093*   9094*   9100*   9102*   9114*   9115*   9116*   9132*
                         9133*   9148*   9149*   9176*   9177*   9230*   9243*   9254*   9255    9267*   9272*   9275    9286*
                         9294*   9297    9323*   9326*   9327*   9330    9342*   9343*   9346    9353*   9705*   9711*   9712*
                         9713*   9714*   9717*   9718    9765*   9770*   9773    9782*   9795*   9800    9811*   9812*   9813*
                         9815    9824*   9825*   9827    9943*   9955*   9968*   9973    9992*  10006*  10007*  10034*  10047*
                        10071*  10072*  10086*  10099*  10100*  10115*  10116*  10203*  10204   10213*  10229*  10231   10240*
                        10253*  10254   10263*  10275*  10276   10285*  10298*  10299   10339*  10340*  10353*  10354*  10372*
                        10373*  10386*  10387*  10400*  10401*  10414*  10415*  10428*  10429*  10480*  10481*  10494*  10495*
                        10510*  10511*  10527*  10528*  10548*  10549*  10565*  10566*  10579*  10580*  10604*  10605*  10620*
                        10622*  10634*  10635*  10650*  10651*  10667*  10668*  10685*  10686*  10699*  10700*  10715*  10716*
                        10752*  10753   10764*  10765   10774*  10805*  10806   10817*  10818   10826*  10885*  10886*  10901*
                        10902*  10922*  10923*  10942*  10943*  10971*  10972*  10990*  10991*  11004*  11005*  11018*  11019*
                        11029*  11042*  11045   11059*  11060*  11092*  11093*  11108*  11109*  11122*  11123*  11140*  11141*
                        11153*  11154*  11168*  11169*  11177*  11208*  11209*  11221*  11222*  11234*  11235*  11247*  11248*
```

```
                    11269* 11270* 11294* 11295* 11309* 11310* 11322* 11323* 11335* 11336* 11348* 11349* 11369*
                    11370* 11386* 11387* 11402* 11403* 11467* 11468* 11483* 11499* 11500* 11519* 11520* 11533*
                    11534* 11592* 11607* 12557
$GET42  033146       7783#
$GTSWR  061754      12218#  12448
$HD   = 000001        811     812
$HIBTS  001100       1300#
$HIOCT  062724      12400*  12405#
$ICNT   001120       1318#  12013*  12014   12016*  12027
$ILLUP  063174      12460   12476   12493#
$INTAG  001151       1332#  12215*  12234   12254   12367
$ITEMB  001130       1322#   7832   12083*  12091   12112
$LF     001220       1354#  11953   12112   12350   12360
$LFLG   063455      12543*  12549#
$LPADR  001122       1319#   3252*   3528*  11995*  12004*  12020*  12025   12027
$LPERR  001124       1320#   3253*   3529*  12004   12021*  12027   12102
$MADR1  001254       1391#
$MADR2  001260       1395#
$MADR3  001264       1398#
$MADR4  001270       1401#
$MAIL   001222       1301    1305    1364#   3270    3285    3506    3534    3574    3618    3656    3824    3992    4162
                     4322    4510    4678    4838    5026    5185    5344    5503    5668    5813    5958    6103    6188
                     6270    6417    6564    6710    6885    7038    7176    7314    7550   11897   12019   12089
$MAMS1  001252       1385#
$MAMS2  001256       1393#
$MAMS3  001262       1396#
$MAMS4  001266       1399#
$MBADR  001102       1301#
$MFLG   063454      12503*  12509   12544*  12548#
$MNEW   062612      12221   12365#
$MSGAD  001236       1371#  12519*  12522
$MSGLG  001240       1372#  12524*
$MSGTY  001222       1365#  12517   12525*  12537   12541*
$MSWR   062601      12218   12363#
$MTYP1  001253       1386#
$MTYP2  001257       1394#
$MTYP3  001263       1397#
$MTYP4  001267       1400#
$MXCNT  061072      12017   12027#
$NULL   001170       1340#  11924   11953
$NWTST= 000001       3523#   3525    3564#   3566    3608#   3610    3647#   3815#   3983#   4153#   4313#   4501#   4669#
                     4829#   5017#   5176#   5335#   5494#   5659#   5804#   5949#   6094#   6179#   6261#   6408#   6555#
                     6701#   6876#   7029#   7167#   7305#   7541#
$OCNT   060270      11829*  11858*  11871#
$OMODE  060272      11824*  11828*  11833   11836*  11847*  11873#
$OVER   061056      11971   11996   12005   12015   12024#
$PASS   001230       1368#   3270*   7759*   7760*   7771    7793   12011   12028
$PASTM  001106       1303#
$POWER  063202      12491   12496#
$PWRDN  063034       3246   12460#  12488
$PWRMG  063170      12491#
$PWRUP  063106      12470   12476#
$QUES   001216       1352#  11953   12112   12272   12343   12360
$RDCHR  062226      12285#  12451
$RDDEC= ****** U    12454
$RDLIN  062316      12308#  12452
```

```
$RDOCT  062624          12378#  12453
$RDSZ = 000010          12301#
$RESRE  057510          11694#  12455
$RTNAD  033170           7792#
$R2A  = ****** U        12456
$SAVRE  057452          11678#  12454
$SAVR6  063200          12469*  12477   12478*  12479*  12495#
$SCOPE  060556           3240   11967#
$SETUP= 000137           1254#   3239    3240    3242    3244    3246    3248    3249    3250    3252    3279    3282    3503
                         7757   11968   12073   12099   12107   12155   12160   12161   12191   12367
$STUP = 177777           1254#
$SVLAD  061022          11979   12018#
$SVPC = 000200           1270#   1275
$SWR  = 167400            800#    811     816     817     818     819     820     821     822     823    1349    1350    1351
                         3249    3250    3252    3253    3534    3574    3618    3656    3824    3992    4162    4322    4510
                         4678    4838    5026    5185    5344    5503    5668    5813    5958    6103    6188    6270    6417
                         6564    6710    6885    7038    7176    7314    7550    7752    7758    7785    7791    7793   11959
                        11960   11961   11962   11963   11970   11982   11984   11985   11998   11999   12000   12007   12008
                        12009   12021   12024   12027   12064   12065   12066   12067   12068   12077   12084   12096   12100
                        12112   12492
$SWREG  001244           1376#   3273
$SWRMK= 000000            823    824    11963   11964   11988
$SWOBT  061074          11994   12028#
$TESTN  001226           1367#   3534*   3574*   3618*   3656*   3824*   3992*   4162*   4322*   4510*   4678*   4838*   5026*
                         5185*   5344*   5503*   5668*   5813*   5958*   6103*   6188*   6270*   6417*   6564*   6710*   6885*
                         7038*   7176*   7314*   7550*   7824   12019*
$TIMES  001206           1349#   3249*   7758*  12007*  12014   12017*  12027
$TKB    001162           1337#  12115   12136   12147   12172   12200   12227
$TKCNT  061366          12116#  12131*  12161   12178*  12292   12294*
$TKINT  061376           3343    3516   12131#  12152   12213
$TKQEN= 061375          12120#  12186   12297
$TKQIN  061370          12117#  12132*  12133   12184*  12185*  12186   12188*
$TKQOU  061372          12118#  12133*  12295   12296*  12297   12299*
$TKQSR  061374          12119#  12132   12188   12299
$TKS    001160           1336#  12115   12137*  12168*  12170   12176*  12198   12214*  12224   12236*  12256*
$TKSRV  061446          12134   12147#
$TMP0   001174           1344#   3588*   3591    3666*   3834*   4002*   4172*   4332*   4520*   4688*   4848*   5036*   5195*
                         5354*   5513*   5678*   5823*   5968*   6280*   6427*   6574*   6720*   6895*   7048*   7186*   7325*
                         7461*   7480*   7561*   7677*   8655    8762*   8851*   8924*   9879*   9880   12557
$TMP1   001176           1345#   3353*   3354    3355    3358    3367*   3368    3369    3377*   3378    3381    3391*   3393
                         3397    3398*   3399*   3401    3407*   3409    3413    3414*   3415*   3416*   3417*   3418*   3419*
                         3420*   3422    3428*   3430    3435*   3436*   3437*   3438*   3439*   3440    3448*   3449    3450
                         3457*   3458    3459    3461    3463    3467    3587*   3595    3641*   3667*   3835*   4003*   4173*
                         4333*   4521*   4689*   4849*   5037*   5196*   5355*   5514*   5679*   5824*   5969*   6281*   6428*
                         6575*   6721*   6896*   7049*   7187*   7326*   7462*   7481*   7562*   7678*   8656    8852*   8853
                         8925*   8926    9878*   9885   11576*  11577*  11589   11591   12557
$TMP2   001200           1346#   3642*   7463*   7482*   7679*  11579*  11580*  11603   11606   12557
$TMP3   001202           1347#
$TMP4   001204           1348#
$TN   = 000036            811#   3523    3534#   3564    3574#   3608    3618#   3647    3656#   3815    3824#   3983    3992#
                         4153    4162#   4313    4322#   4501    4510#   4669    4678#   4829    4838#   5017    5026#   5176
                         5185#   5335    5344#   5494    5503#   5659    5668#   5804    5813#   5949    5958#   6094    6103#
                         6179    6188#   6261    6270#   6408    6417#   6555    6564#   6701    6710#   6876    6885#   7029
                         7038#   7167    7176#   7305    7314#   7541    7550#  11989   12029
$TPB    001166           1339#  11942*  11953
$TPFLG  001173           1343#  11891   11953
```

```
$TPS    001164          1338#  11940   11953
$TRAP   062726          3244   12414#
$TRAP2  062766         12429#  12440
$TRP  = 000016         12433#  12442#  12443#  12444#  12445#  12446#  12447#  12448   12449#  12450   12451#  12452#  12453#
                       12454#  12455#  12456#
$TRPAD  063000         12423   12440#
$TSTM   001104          1302#
$TSTNM  001116          1316#   7757*  11958   11991*  12018*  12019   12024   12028   12076   12112
$TTYIN  062552         12310   12311   12323   12341   12355   12359#
$TYPBN  057546         11715#  12446
$TYPDS  057622         11742#  12445
$TYPE   060274         11891#  12433   12441   12530
$TYPEC  060506         11921   11928   11935   11940#  11941   12258
$TYPEX  060554         11946   11948   11951#
$TYPOC  060072         11827#  12442
$TYPON  060106         11826   11829#  12444
$TYPOS  060046         11822#  12443
$UNIT   001234          1370#   3521*   7818    9075
$UNITM  001110          1304#
$USWR   001246          1377#
$VECT1  001272          1402#   3398    3413*   3414    3440*
$VECT2  001274          1403#
$XTSTR  060574         11973#
$$GET4= 000000          7785#
$$SW08= 000036         12028#  12029   12030#  12031#  12032#  12033#  12034#  12035#  12036#  12037#  12038#  12039#  12040#
                       12041#  12042#  12043#  12044#  12045#  12046#  12047#  12048#  12049#  12050#  12051#  12052#  12053#
                       12054#  12055#  12056#  12057#  12058#
$OFILL  060271         11823*  11827*  11837   11872#
$40CAT= ****** U       11970   12086
.     = 106117          1257#   1261#   1270    1271#   1273#   1275#   1280#   1283#   1289    1290#   1292#   1294#   1306
                        1313#   1355    1478#   1494#   1499#   3237    3252    3253    7770#   7793    7794#   11648   11660#
                       11796#  11953   12027   12028   12112   12115   12119#  12120   12121#  12359#  12360   12367#  12472
                       12494   12551#  12557#
.$ASTA= ****** U       12503   12506
.$X   = 001100          1289#   1294

. ABS.  106117     000

 ERRORS DETECTED:  0

 DSKZ:CZRMDC,DSKZ:CZRMDC.SEQ/SOL/CRF/DOC/NL:MC:MD:CND:TOC/LI:ME=DSKZ:CZRMDC.P11
 RUN-TIME: 70 66 3 SECONDS
 RUN-TIME RATIO: 633/140=4.5
 CORE USED:  33K  (65 PAGES)

 DOCUMENT PAGES:  307
```