RM03

FUNCTIONAL TEST PART 3

MD-11-DZRME-A

EP-DZRME-A-DL-A

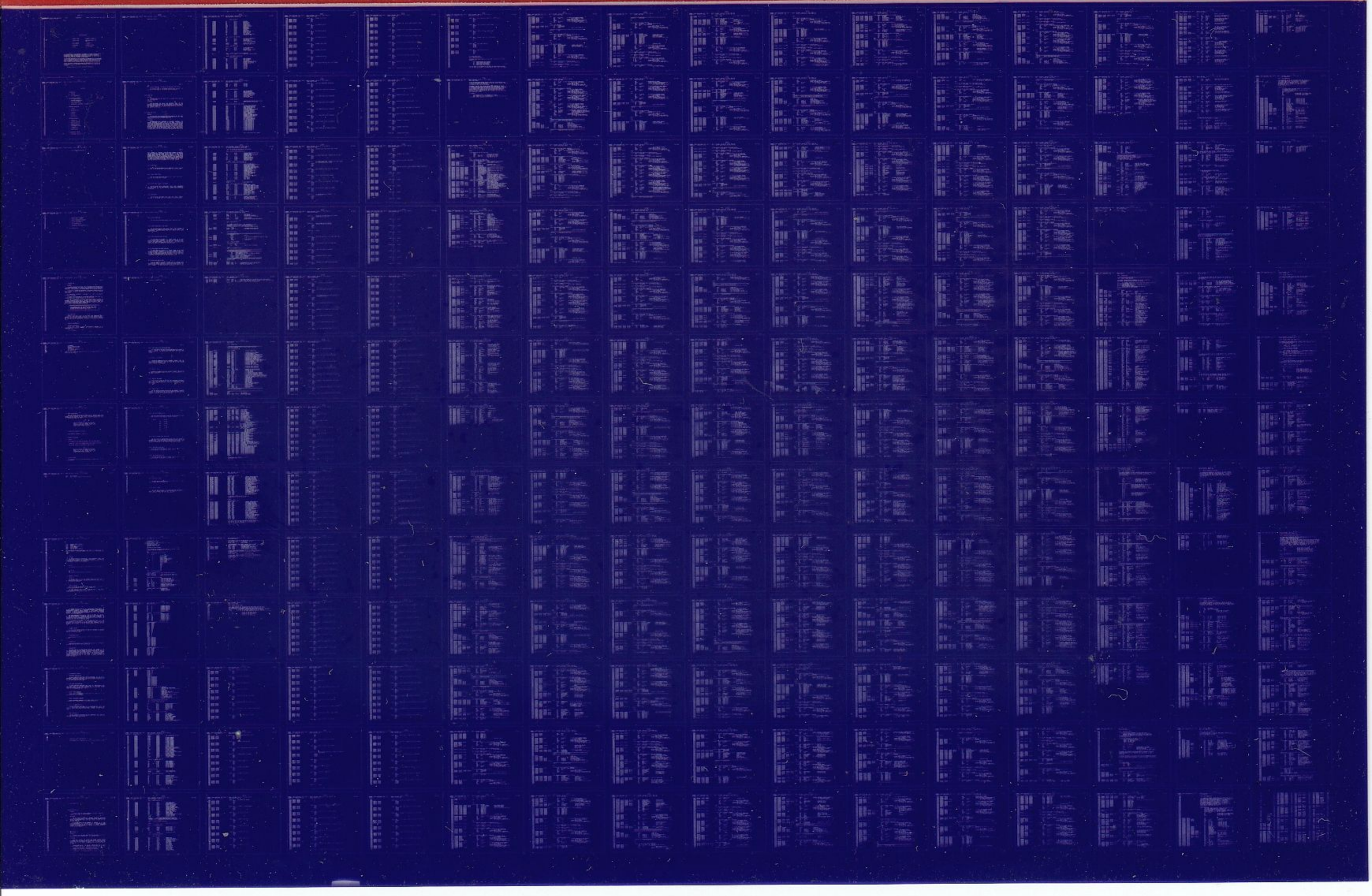
COPYRIGHT © 1977

OCT 1977

digital

FICHE 1 OF 2

MADE IN USA

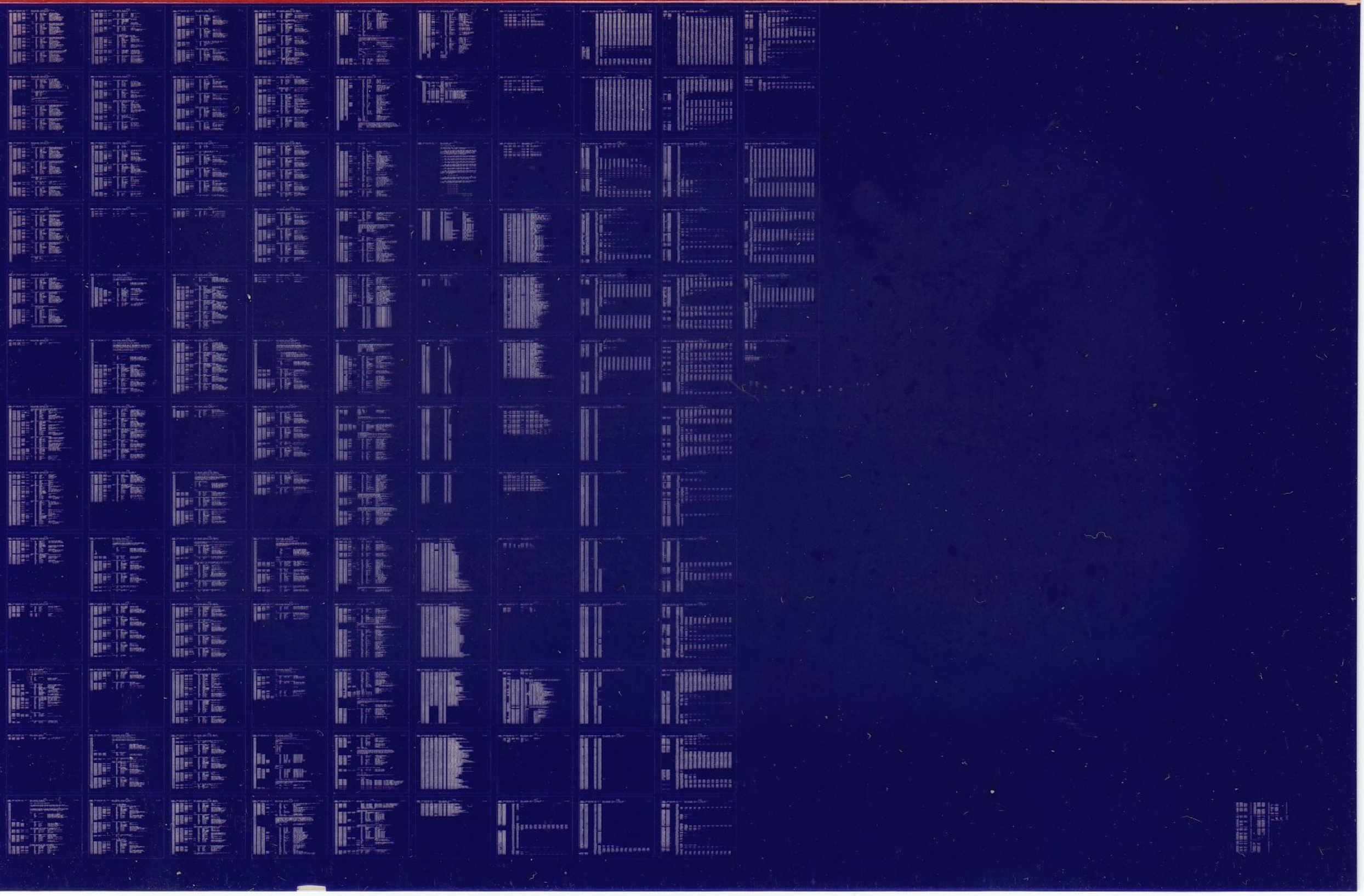


RM03

FUNCTIONAL TEST PART 3
MD-11-DZRME-A

EP-DZRME-A-DL-A
COPYRIGHT © 1977
FICHE 2 OF 2

OCT 1977
digital
MADE IN USA



```
.REM    \
```

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZRME-A-D
PRODUCT NAME:	RM03 FUNCTIONAL TEST, PART 3
DATE CREATED:	1 AUGUST 77
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	DOUG RIIKONEN

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS MANUAL.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURSHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1977, 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

109 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.

```
**********************************************************
*                                                        *
* NOTE: PARTS 2 AND 3 OF THE FUNCTIONAL TEST LEAVE       *
* 2 HEADER ERRORS ON THE MEDIA. THE PACK SHOULD BE       *
* REFORMATTED AFTER RUNNING THESE TESTS.                 *
*                                                        *
**********************************************************
```

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,

                                    WHERE THE UNIT UNDER TEST CONSISTS OF ONE TO EIGHT RM03 ADAPTERS, DISK
                                    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, DZRMJ-A

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.

```
247                             SW15    HALT ON ERROR
248                             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. PROCESSOR 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

```
 474                                    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 DATA TESTS

PURPOSE:

TO TEST WRITE DATA AND READ 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. THEN, THE TEST FORMATS THE SECTOR BEING USED, WHICH MAY VARY FROM THE PROGRAM LISTING, BECAUSE SECTORS ARE SUBSTITUTED DURING RUN TIME IF THE SELECTED SECTOR IS LISTED IN THE BAD BLOCK TABLE OF THE LAST TRACK. FOLLOWING THAT, THE TEST PERFORMS ANY EXPLICIT SEEKS REQUIRED FOR THE CONDITIONS OF THE TEST. REGISTERS ARE PRESET AND THE WRITE

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 DATA PORTION OF THE TEST, VERIFYING THE SAME TYPE OF ERRORS AS IN THE WRITE COMMAND.

WRITE, READ ZEROS

THE TEST WRITES AND READS AN ALL ZEROS DATA FIELD, CAUSING THE DRIVE TO USE NORMAL WRITE GATE THOUGHOUT THE WRITE PROCESS.

WRITE, WRITE CHECK ZEROS

THE TEST WRITES AND WRITE CHECKS AN ALL ZEROS DATA FIELD, VERIFYING THAT THERE ARE NO ERRORS.

WRITE, WRITE CHECK ZEROS W/ WCE ERROR

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

WRITE, READ ONES

THE TEST WRITES AND READS AN ALL ONES DATA FIELD, CAUSING THE DRIVE TO USE NORMAL WRITE GATE THOUGHOUT THE WRITE PROCESS.

WRITE, WRITE CHECK ONES

THE TEST WRITES AND WRITE CHECKS AN ALL ONES DATA FIELD.

WRITE, WRITE CHECK ONES W/ WCE ERROR

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

WRITE, WRITE CHECK MULTIPLE SECTORS

THE TEST SEEKS TO CYLINDER 0 TO INSURE THERE IS NO HEAD MOTION DURING DATA TRANSFER. THE WRITE DATA COMMAND FOLLOWS, WITH THE WORD COUNT EQUAL TO MULTIPLE SECTORS. THE WORD COUNT DURING THE READ DATA PORTION OF THE TEST IS SET FOR ALL OF THE FIRST SECTOR AND THE 1 WORD OF THE SECOND SECTOR.

WRITE, WRITE CHECK WITH HEAD SWITCHING

THE TEST SEEKS TO CYLINDER 0 TO INSURE THERE IS NO HEAD MOTION DURING DATA TRANSFER. THE WRITE 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 DATA COMMAND USES THE SAME WORD COUNT AND STARTING SECTOR.

WRITE, READ WITH IMPLIED SEEK

THIS TEST SEEKS TO THE LAST CYLINDER PRIOR TO WRITING DATA ON CYLINDER 0, TRACK 1, SECTOR 1. 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

635 IS REPEATED FOR READ DATA.

WRITE, WRITE CHECK 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 DATA COMMAND.

WRITE, READ W/ HCE ERROR

A SECTOR IS FORMATTED WITH AN INCORRECT HEADER, THEN THE TEST WRITES AND READS DATA FROM THE SECTOR AND VERIFIES THAT THE CORRECT ERROR IS DETECTED. EACH BIT POSITION OF BOTH HEADER WORDS IS TESTED IN THIS MANNER.

WRITE, READ W/ HCI

A SECTOR IS FORMATTED WITH AN INCORRECT HEADER, THEN WRITTEN AND READ WITH HEADER COMPARE INHIBITED. THE TEST VERIFIES THAT NO ERROR IS DETECTED.

WRITE, READ W/ IVC ERROR

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

WRITE, READ W/ BAI

THE TEST WRITES A SINGLE SECTOR WITH "BUS ADDRESS INCREMENT INHIBIT" SET AND VERIFIES THAT THE BUS ADDRESS REGISTER DOES NOT INCREMENT. THE SAME SECTOR IS READ WITH "BAI" RESET AND THE TEST CHECKS TO SEE THAT ALL THE DATA IS THE SAME.

WRITE, READ EACH CURRENT LEVEL

THE TEST WRITES AND READS ON EACH OF THE FOLLOWING CYLINDERS IN ORDER TO WRITE AT EACH POSSIBLE CURRENT THRESHOLD.

CYLINDER	0	110 MA
"	128	104 MA
"	256	97 MA
"	384	91 MA
"	512	84 MA
"	640	77 MA
"	768	70 MA

WRITE, READ W/CURRENT LEVEL SWITCHING

THE TEST WRITES 2 SECTORS STARTING WITH THE LAST SECTOR OF CYLINDER 383. THE FIRST SECTOR IS WRITTEN AT ONE CURRENT LEVEL AND THE SECOND SECTOR IS WRITTEN AT ANOTHER CURRENT LEVEL. BOTH SECTORS ARE READ AND THE TEST VERIFIES THERE ARE NO ERRORS.

WRITE, READ EARLY PEAK SHIFT

THIS TEST WRITES AND READS A SINGLE SECTOR USING A DATA PATTERN DESIGNED TO INTRODUCE EARLY PEAK SHIFT.

WRITE, READ MIXED PEAK SHIFT

THIS TEST WRITES AND READS A SINGLE SECTOR USING A DATA PATTERN DESIGNED TO INTRODUCE MIXED PEAK SHIFT.

PAGE 38

WRITE, READ EACH TRACK

THE TEST WRITES AND READS WITH EACH HEAD USING A MIX OF EARLY, NORMAL AND LATE WRITE GATES.

WRITE, READ W/ ABORT

THE TEST SETS AN ERROR IN THE ERROR REGISTER AND EXECUTES A WRITE DATA COMMAND VERIFYING THAT "PIP" REMAINS INACTIVE. THE SAME PROCEDURE IS USED FOR READ DATA COMMAND.

```
                                ;PROGRAM REVISION #001

                                .TITLE  DZRMEA - RM03 FUNCTIONAL TEST, PART 3
                                ;*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
```

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

                        ;*PRIORITY LEVEL DEFINITIONS
                        PR0=    0               ;;PRIORITY LEVEL 0
                        PR1=    40              ;;PRIORITY LEVEL 1
                        PR2=    100             ;;PRIORITY LEVEL 2
                        PR3=    140             ;;PRIORITY LEVEL 3
                        PR4=    200             ;;PRIORITY LEVEL 4
                        PR5=    240             ;;PRIORITY LEVEL 5
                        PR6=    300             ;;PRIORITY LEVEL 6
                        PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                        SW15=   100000
                        SW14=   40000
                        SW13=   20000
                        SW12=   10000
                        SW11=   4000
                        SW10=   2000
                        SW09=   1000
                        SW08=   400
                        SW07=   200
                        SW06=   100
                        SW05=   40
                        SW04=   20
                        SW03=   10
                        SW02=   4
                        SW01=   2
                        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)
                        BIT15=  100000
                        BIT14=  40000
                        BIT13=  20000
                        BIT12=  10000
                        BIT11=  4000
                        BIT10=  2000
                        BIT09=  1000
                        BIT08=  400
                        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
```

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

```
                                ;RMER1  ERROR REGISTER #1

            100000              DCK     =       BIT15           ;DATA CHECK ERROR
            040000              UNS     =       BIT14           ;DRIVE UNSAFE
            020000              OPI     =       BIT13           ;OPERATION INCOMPLETE
            010000              DTE     =       BIT12           ;DRIVE TIMING ERROR
            004000              WLE     =       BIT11           ;WRITE LOCK ERROR
            002000              IAE     =       BIT10           ;INVALID ADDRESS ERROR
            001000              AOE     =       BIT09           ;ADDRESS OVERFLOW ERROR
            000400              HCRC    =       BIT08           ;HEADER CRC ERROR
            000200              HCE     =       BIT07           ;HEADER COMPARE ERROR
            000100              ECH     =       BIT06           ;ECC "HARD" ERROR
            000040              WCF     =       BIT05           ;WRITE CLOCK FAILURE
            000020              FER     =       BIT04           ;FORMAT ERROR
            000010              PAR     =       BIT03           ;PARITY ERROR
            000004              RMR     =       BIT02           ;REGISTER MODIFICATION REFUSED
            000002              ILR     =       BIT01           ;ILLEGAL REGISTER
            000001              ILF     =       BIT00           ;ILLEGAL FUNCTION

            115760              NDTMSK  =       DCK!DTE!WLE!AOE!HCRC!HCE!ECH!WCF!FER
                                ;"NDTMSK" IS USED TO MASK ERROR REGISTER 1 DURING NON - DATA
                                ;COMMANDS, I.E., HOUSEKEEPING AND POSITIONING COMMANDS

                                ;RMAS   ATTENTION SUMMARY REGISTER

            000377              ATNMSK  =       377             ;MASK FOR ATTENTION BITS

                                ;RMLA   LOOK AHEAD REGISTER

            002000              SC4     =       BIT10           ;SECTOR COUNT = 16
            001000              SC3     =       BIT09           ;SECTOR COUNT = 8
            000400              SC2     =       BIT08           ;SECTOR COUNT = 4
            000200              SC1     =       BIT07           ;SECTOR COUNT = 2
            000100              SC0     =       BIT06           ;SECTOR COUNT = 1

            003700              SCTMSK  =       003700          ;SECTOR COUNT MASK

                                ;RMMR   MAINTENANCE REGISTER

                                ;       WRITE ONLY BITS

            100000              DBCK    =       BIT15           ;DEBUG CLOCK
            040000              DBEN    =       BIT14           ;DEBUG CLOCK ENABLE
            020000              DEBL    =       BIT13           ;DIAGNOSTIC END OF BLOCK
            010000              DTO     =       BIT12           ;DIAGNOSTIC TIMEOUT
            004000              MCLK    =       BIT11           ;MAINTENANCE CLOCK
            002000              MRD     =       BIT10           ;READ DATA
            001000              MUR     =       BIT09           ;UNIT READY
            000400              MOC     =       BIT08           ;ON CYLINDER
            000200              MSER    =       BIT07           ;SEEK ERROR
            000100              MDF     =       BIT06           ;DRIVE FAULT
            000040              MS      =       BIT05           ;SECTOR PULSE
            000010              MWP     =       BIT03           ;WRITE PROTECT
            000004              MI      =       BIT02           ;INDEX PULSE
            000002              MSC     =       BIT01           ;SECTOR COMPARE
            000001              DMD     =       BIT00           ;DIAGNOSTIC MODE
```

```
1043
1044                            ;       READ ONLY BITS
1045
1046    100000                  OCC     =       BIT15           ;OCCUPIED
1047    040000                  RG      =       BIT14           ;RUN AND GO
1048    020000                  EBL     =       BIT13           ;END OF BLOCK
1049    010000                  REX     =       BIT12           ;EXCEPTION
1050    004000                  ESRC    =       BIT11           ;ENABLE SEARCH
1051    002000                  PLFS    =       BIT10           ;LOOKING FOR SYNC
1052    001000                  ECRC    =       BIT09           ;ENABLE CRC OUT
1053    000400                  PDA     =       BIT08           ;DATA AREA
1054    000200                  PHA     =       BIT07           ;HEADER AREA
1055    000100                  CONT    =       BIT06           ;CONTINUE
1056    000040                  WC      =       BIT05           ;WORD CLOCK
1057    000020                  EECC    =       BIT04           ;ENABLE ECC OUT
1058    000010                  MWD     =       BIT03           ;WRITE DATA BIT
1059    000004                  LS      =       BIT02           ;LAST SECTOR
1060    000002                  LST     =       BIT01           ;LAST SECTOR AND TRACK
1061    000001                  DMD     =       BIT00           ;DIAGNOSTIC MODE
1062
1063                            ;RMDT   DRIVE TYPE REGISTER
1064
1065    100000                  NSA     =       BIT15           ;NOT SECTOR ADDRESSED=0
1066    040000                  TAP     =       BIT14           ;TAPE DRIVE = 0
1067    020000                  MOH     =       BIT13           ;MOVING HEAD = 1
1068    004000                  DRQ     =       BIT11           ;DRIVE REQUEST REQUIRED
1069
1070    020024                  SNGPRT  =       020024          ;SINGLE PORT DRIVE TYPE
1071    024024                  DULPRT  =       024024          ;DUAL PORT DRIVE TYPE
1072
1073                            ;RMOF   OFFSET REGISTER
1074
1075    010000                  FMT16   =       BIT12           ;16 BIT WORD FORMAT
1076    004000                  ECI     =       BIT11           ;ECC INHIBIT
1077    002000                  HCI     =       BIT10           ;HEADER COMPARE INHIBIT
1078    000200                  OFD     =       BIT07           ;OFFSET FORWARD
1079
1080
1081                            ;RMDC   DESIRED CYLINDER ADDRESS REGISTER
1082    001777                  CYLMSK  =       1777            ;MASK FOR CYLINDER ADDRESS
1083
1084                            ;RMMR2  MAINTENANCE REGISTER #2
1085
1086                            ;       READ ONLY BITS
1087    100000                  RQA     =       BIT15           ;PORT A REQUEST
1088    040000                  RQB     =       BIT14           ;PORT B REQUEST
1089    020000                  TAG     =       BIT13           ;TAG CONTROL
1090    010000                  TST     =       BIT12           ;COMMAND SEQUENCE TEST BIT
1091    004000                  CC      =       BIT11           ;CONTROL OR CYLINDER TAG
1092    002000                  CH      =       BIT10           ;CONTROL OR HEAD TAG
1093    001000                  BB09    =       BIT09           ;TAG BUS
1094    000400                  BB08    =       BIT08           ;TAG BUS
1095    000200                  BB07    =       BIT07           ;TAG BUS
1096    000100                  BB06    =       BIT06           ;TAG BUS
1097    000040                  BB05    =       BIT05           ;TAG BUS
1098    000020                  BB04    =       BIT04           ;TAG BUS
```

```
000010          BB03    =       BIT03           ;TAG BUS
000004          BB02    =       BIT02           ;TAG BUS
000002          BB01    =       BIT01           ;TAG BUS
000001          BB00    =       BIT00           ;TAG BUS


                ;RMER2  ERROR REGISTER 2

100000          BSE     =       BIT15           ;BAD SECTOR ERROR
040000          SKI     =       BIT14           ;SEEK INCOMPLETE
020000          OPE     =       BIT13           ;OPERATOR PLUG ERROR
010000          IVC     =       BIT12           ;INVALID COMMAND ERROR
004000          LSC     =       BIT11           ;LOSS OF SYSTEM CLOCK
002000          LBC     =       BIT10           ;LOSS OF BIT CLOCK
000200          DVC     =       BIT07           ;DEVICE CHECK
000010          DPE     =       BIT03           ;DATA PARITY ERROR

                .SBTTL  PROGRAM MNEMONICS

100000          MSE     =       BIT15           ;MANUFACTURING DETECTED SECTOR ERROR
040000          USE     =       BIT14           ;USER DETECTED SECTOR ERROR

                .SBTTL  RM03 REGISTER INDEX VALUES

000000          RMCS1   =       00              ;CONTROL STATUS REGISTER
000006          RMDA    =       06              ;DISK ADDRESS REGISTER
000012          RMDS    =       12              ;DRIVE STATUS REGISTER
000014          RMER1   =       14              ;ERROR REGISTER 1
000016          RMAS    =       16              ;ATTENTION SUMMARY REGISTER
000020          RMLA    =       20              ;LOOK AHEAD REGISTER
000024          RMMR1   =       24              ;MAINTENANCE REGISTER
000026          RMDT    =       26              ;DRIVE TYPE REGISTER
000030          RMSN    =       30              ;SERIAL NUMBER REGISTER
000032          RMOF    =       32              ;OFFSET REGISTER
000034          RMDC    =       34              ;DESIRED CYLINDER REGISTER
000036          RMCC    =       36              ;CURRENT CYLINDER REGISTER
000040          RMMR2   =       40              ;MAINTENANCE REGISTER 2
000042          RMER2   =       42              ;ERROR REGISTER 2
000044          RMEC1   =       44              ;ECC POSITION REGISTER
000046          RMEC2   =       46              ;ECC PATTERN REGISTER
000050          ILRG50  =       50              ;ILLEGAL REGISTER 50
000052          ILRG52  =       52              ;ILLEGAL REGISTER 52
000054          ILRG54  =       54              ;ILLEGAL REGISTER 54
000056          ILRG56  =       56              ;ILLEGAL REGISTER 56
000060          ILRG60  =       60              ;ILLEGAL REGISTER 60
000062          ILRG62  =       62              ;ILLEGAL REGISTER 62
000064          ILRG64  =       64              ;ILLEGAL REGISTER 64
000066          ILRG66  =       66              ;ILLEGAL REGISTER 66
000070          ILRG70  =       70              ;ILLEGAL REGISTER 70
000072          ILRG72  =       72              ;ILLEGAL REGISTER 72
000074          ILRG74  =       74              ;ILLEGAL REGISTER 74
000076          ILRG76  =       76              ;ILLEGAL REGISTER 76


000077          IDXMSK  =       77              ;MASK FOR REGISTER INDEX NUMBER
```

```
                                .SBTTL  RH CONTROLLER REGISTER BIT DEFINITIONS

                                ;RMCS1  CONTROL STATUS REGISTER #1

100000                          SC      =       BIT15           ;SPECIAL CONDITION-READ ONLY
040000                          TRE     =       BIT14           ;TRANSFER ERROR
020000                          MCPE    =       BIT13           ;MASSBUS CONTROL BUS PARITY
                                                                ;ERROR-READ ONLY
002000                          PSEL    =       BIT10           ;PORT B SELECT
001000                          A17     =       BIT09           ;ADDRESS EXTENSION
000400                          A16     =       BIT08           ;ADDRESS EXTENSION
000200                          RDY     =       BIT07           ;READY-READ ONLY
000100                          IE      =       BIT06           ;INTERRUPT ENABLE

                                ;RMCS2  RH CONTROL STATUS REGISTER #2

100000                          DLT     =       BIT15           ;DATA LATE-READ ONLY
040000                          WCE     =       BIT14           ;WRITE CHECK ERROR-READ ONLY
020000                          UPE     =       BIT13           ;UNIBUS PARITY ERROR
010000                          NED     =       BIT12           ;NONEXISTANT DRIVE-READ ONLY
004000                          NEM     =       BIT11           ;NONEXISTANT MEMORY-READ ONLY
002000                          PGE     =       BIT10           ;PROGRAM ERROR-READ ONLY
001000                          MXF     =       BIT09           ;MISSED TRANSFER
000400                          MDPE    =       BIT08           ;MASSBUS DATA BUS PARITY
                                                                ;ERROR-READ ONLY
000200                          OR      =       BIT07           ;OUTPUT READY-READ ONLY
000100                          IR      =       BIT06           ;INPUT READY-READ ONLY
000040                          CLR     =       BIT05           ;CONTROLLER CLEAR
000020                          PAT     =       BIT04           ;PARITY TEST
000010                          BAI     =       BIT03           ;UNIBUS ADDRESS INCREMENT
                                                                ;INHIBIT
000004                          U2      =       BIT02           ;UNIT SELECT
000002                          U1      =       BIT01           ;UNIT SELECT
000001                          U0      =       BIT00           ;UNIT SELECT

                                ;UNIT SELECT MASK
000007                          UNTMSK  =       7               ;UNIT SELECT MASK

                                ;RMCS3  RH70 CONTROL STATUS REGISTER #3
100000                          APE     =       BIT15           ;ADDRESS PARITY ERROR
040000                          DPEHI   =       BIT14           ;DATA PARITY ERROR HIGH WORD
020000                          DPELO   =       BIT13           ;DATA PARITY ERROR LOW WORD
010000                          WCEHI   =       BIT12           ;WRITE CHECK ERROR HIGH WORD
004000                          WCELO   =       BIT11           ;WRITE CHECK ERROR LOW WORD
002000                          DBL     =       BIT10           ;DOUBLE WORD TRANSFER
000100                          IE      =       BIT06           ;INTERRUPT ENABLE
000010                          IPCK3   =       BIT03           ;INVERT PARITY CHECK
000004                          IPCK2   =       BIT02           ;INVERT PARITY CHECK
000002                          IPCK1   =       BIT01           ;INVERT PARITY CHECK
000001                          IPCK0   =       BIT00           ;INVERT PARITY CHECK
                                .SBTTL  RH CONTROLLER REGISTER INDEX VALUES

000000                          RMCS1   =       00              ;CONTROL, STATUS REGISTER
000002                          RMWC    =       02              ;WORD COUNT REGISTER
000004                          RMBA    =       04              ;BUS ADDRESS REGISTER
000010                          RMCS2   =       10              ;CONTROLLER STATUS REGISTER
```

```
                    000022                  RMDB    =       22              ;DATA BUFFER
                    000050                  RMBAE   =       50              ;BUS ADDRESS EXTENSION
                    000052                  RMCS3   =       52              ;CONTROL STATUS REGISTER #3

                    176700                  ABASE   =       176700          ;UNIBUS ADDRESS
                    120254                  AVECT1  =       120254          ;UNIBUS VECTOR ADDRESS AND PRIORITY


                                            .SBTTL  TRAP CATCHER

                    000000                          .=0
                                            ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                            ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                            ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                    000174                  .=174
000174  000000                             DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                             SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER


                                            .SBTTL  ACT11 HOOKS

                                            ;;**************************************************************
                                            ;HOOKS REQUIRED BY ACT11
                    000200                          $SVPC=.                 ;SAVE PC
                    000046                          .=46
000046  036436                              $ENDAD                          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                    000052                          .=52
000052  000000                                  .WORD   0                   ;;2)SET LOC.52 TO ZERO
                    000200                          .=$SVPC                 ;; RESTORE PC


                                            .SBTTL  STARTING ADDRESS

                                            ;THE PROGRAM STARTS AT LOCATION 200
                    000200                  .       =       200
000200  000137  005322                              JMP     START           ;JUMP TO START OF PROGRAM


                    001100                  .=1100
                                            .SBTTL  APT PARAMETER BLOCK

                                            ;;**************************************************************
                                            ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                            ;;**************************************************************
                    001100                          .$X=.   ;;SAVE CURRENT LOCATION
                    000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                                  200     ;;FOR APT START UP
                    000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001100                                  $APTHDR ;;POINT TO APT HEADER BLOCK
                    001100                          .=.$X   ;;RESET LOCATION COUNTER
                                            ;;**************************************************************
                                            ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                            ;INTERFACE SPEC.

001100                                      $APTHD:
001100  000000                             $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001102  001222                             $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001104  000001                             $TSTM:  .WORD   1       ;;RUN TIM OF LONGEST TEST
```

```
                        APT PARAMETER BLOCK

  1267  001106  000002  $PASTM: .WORD   2        ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
  1268  001110  000002  $UNITM: .WORD   2        ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
  1269  001112  000042          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
  1270          001114  TAGADR  =       .
```

```
                        .SBTTL  COMMON TAGS

                        ;;*****************************************************************
                        ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                        ;*USED IN THE PROGRAM.

001114                          .=TAGADR
001114                  $CMTAG:                         ;;START OF COMMON TAGS
001114  000000                  .WORD   0
001116  000                     $TSTNM: .BYTE   0       ;;CONTAINS THE TEST NUMBER
001117  000                     $ERFLG: .BYTE   0       ;;CONTAINS ERROR FLAG
001120  000000                  $ICNT:  .WORD   0       ;;CONTAINS SUBTEST ITERATION COUNT
001122  000000                  $LPADR: .WORD   0       ;;CONTAINS SCOPE LOOP ADDRESS
001124  000000                  $LPERR: .WORD   0       ;;CONTAINS SCOPE RETURN FOR ERRORS
001126  000000                  $ERTTL: .WORD   0       ;;CONTAINS TOTAL ERRORS DETECTED
001130  000                     $ITEMB: .BYTE   0       ;;CONTAINS ITEM CONTROL BYTE
001131  001                     $ERMAX: .BYTE   1       ;;CONTAINS MAX. ERRORS PER TEST
001132  000000                  $ERRPC: .WORD   0       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001134  000000                  $GDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'GOOD' DATA
001136  000000                  $BDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'BAD' DATA
001140  000000                  $GDDAT: .WORD   0       ;;CONTAINS 'GOOD' DATA
001142  000000                  $BDDAT: .WORD   0       ;;CONTAINS 'BAD' DATA
001144  000000                          .WORD   0       ;;RESERVED--NOT TO BE USED
001146  000000                          .WORD   0
001150  000                     $AUTOB: .BYTE   0       ;;AUTOMATIC MODE INDICATOR
001151  000                     $INTAG: .BYTE   0       ;;INTERRUPT MODE INDICATOR
001152  000000                          .WORD   0
001154  177570                  SWR:    .WORD   DSWR    ;;ADDRESS OF SWITCH REGISTER
001156  177570                  DISPLAY: .WORD  DDISP   ;;ADDRESS OF DISPLAY REGISTER
001160  177560                  $TKS:   177560          ;;TTY KBD STATUS
001162  177562                  $TKB:   177562          ;;TTY KBD BUFFER
001164  177564                  $TPS:   177564          ;;TTY PRINTER STATUS REG. ADDRESS
001166  177566                  $TPB:   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
001170  000                     $NULL:  .BYTE   0       ;;CONTAINS NULL CHARACTER FOR FILLS
001171  002                     $FILLS: .BYTE   2       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001172  012                     $FILLC: .BYTE   12      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001173  000                     $TPFLG: .BYTE   0       ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001174  000000                  $TMP0:  .WORD   0       ;;USER DEFINED
001176  000000                  $TMP1:  .WORD   0       ;;USER DEFINED
001200  000000                  $TMP2:  .WORD   0       ;;USER DEFINED
001202  000000                  $TMP3:  .WORD   0       ;;USER DEFINED
001204  000000                  $TMP4:  .WORD   0       ;;USER DEFINED
001206  000000                  $TIMES: 0               ;;MAX. NUMBER OF ITERATIONS
001210  000000                  $ESCAPE:0               ;;ESCAPE ON ERROR ADDRESS
001212  177607  000377           $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001216  077                     $QUES:  .ASCII  /?/     ;;QUESTION MARK
001217  015                     $CRLF:  .ASCII  <15>    ;;CARRIAGE RETURN
001220  000012                  $LF:    .ASCIZ  <12>    ;;LINE FEED
                        ;;*****************************************************************
                        .SBTTL  APT MAILBOX-ETABLE

                        ;;*****************************************************************
                        .EVEN
001222                  $MAIL:                          ;;APT MAILBOX
001222  000000                  $MSGTY: .WORD   AMSGTY   ;;MESSAGE TYPE CODE
001224  000000                  $FATAL: .WORD   AFATAL   ;;FATAL ERROR NUMBER
```

```
1327  001226  000000                  $TESTN: .WORD   ATESTN  ;;TEST NUMBER
1328  001230  000000                  $PASS:  .WORD   APASS   ;;PASS COUNT
1329  001232  000000                  $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
1330  001234  000000                  $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
1331  001236  000000                  $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
1332  001240  000000                  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
1333  001242                          $ETABLE:                ;;APT ENVIRONMENT TABLE
1334  001242     000                  $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
1335  001243     000                  $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
1336  001244  000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
1337  001246  000000                  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
1338  001250  000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
1339                                  ;*                      BITS 15-11=CPU TYPE
1340                                  ;*                           11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1341                                  ;*                           11/70=06,PDQ=07,Q=10
1342                                  ;*                      BIT 10=REAL TIME CLOCK
1343                                  ;*                      BIT  9=FLOATING POINT PROCESSOR
1344                                  ;*                      BIT  8=MEMORY MANAGEMENT
1345  001252     000                  $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
1346  001253     000                  $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
1347                                  ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
1348                                  ;*                           900 NSEC CORE=001
1349                                  ;*                           300 NSEC BIPOLAR=002
1350                                  ;*                           500 NSEC MOS=003
1351  001254  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
1352                                  ;*                      MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
1353  001256     000                  $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
1354  001257     000                  $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
1355  001260  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
1356  001262     000                  $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
1357  001263     000                  $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
1358  001264  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
1359  001266     000                  $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
1360  001267     000                  $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
1361  001270  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
1362  001272  120254                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1363  001274  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1364  001276  176700                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
1365  001300  000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
1366  001302  000000                  $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
1367  001304  000000                  $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
1368  001306  000000                  $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
1369  001310  000000                  $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
1370  001312  000000                  $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
1371  001314  000000                  $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
1372  001316  000000                  $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
1373  001320  000000                  $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
1374  001322  000000                  $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
1375  001324  000000                  $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
1376  001326                          $ETEND:
1377                                  .MEXIT
1378
1379                                  ;THE REGISTER INPUT BUFFER IS USED FOR
1380                                  ;STORING DRIVE STATUS
1381
1382  001326                          GETBUF:
```

```
                                ;REGISTER INPUT BUFFER
001326  000000                  RMCS1I: .WORD                   ;CONTROL,STATUS REGISTER
001330  000000                  RMWCI:  .WORD                   ;WORD COUNT REGISTER
001332  000000                  RMBAI:  .WORD                   ;BUS ADDRESS REGISTER
001334  000000                  RMDAI:  .WORD                   ;DISK ADDRESS REGISTER
001336  000000                  RMCS2I: .WORD                   ;CONTROLLER STATUS REGISTER
001340  000000                  RMDSI:  .WORD                   ;DRIVE STATUS REGISTER
001342  000000                  RMER1I: .WORD                   ;ERROR REGISTER 1
001344  000000                  RMASI:  .WORD                   ;ATTENTION SUMMARY REGISTER
001346  000000                  RMLAI:  .WORD                   ;LOOK AHEAD REGISTER
001350  000000                  RMDBI:  .WORD                   ;DATA BUFFER
001352  000000                  RMMR1I: .WORD                   ;MAINTENANCE REGISTER #1
001354  000000                  RMDTI:  .WORD                   ;DRIVE TYPE REGISTER
001356  000000                  RMSNI:  .WORD                   ;SERIAL NUMBER REGISTER
001360  000000                  RMOFI:  .WORD                   ;OFFSET REGISTER
001362  000000                  RMDCI:  .WORD                   ;DESIRED CYLINDER REGISTER
001364  000000                  RMCCI:  .WORD                   ;CURRENT CYLINDER REGISTER
001366  000000                  RMMR2I: .WORD                   ;MAINTENANCE REGISTER #2
001370  000000                  RMER2I: .WORD                   ;ERROR REGISTER 2
001372  000000                  RMEC1I: .WORD                   ;ECC POSITION REGISTER
001374  000000                  RMEC2I: .WORD                   ;ECC PATTERN REGISTER

                                ;THE REGISTER OUTPUT BUFFER IS USED FOR
                                ;ASSEMBLING DATA GOING TO REGISTER

001376                          PUTBUF:

                                ;REGISTER OUTPUT BUFFER
001376  000000                  RMCS10: .WORD                   ;CONTROL,STATUS REGISTER
001400  000000                  RMWC0:  .WORD                   ;WORD COUNT REGISTER
001402  000000                  RMBA0:  .WORD                   ;BUS ADDRESS REGISTER
001404  000000                  RMDA0:  .WORD                   ;DISK ADDRESS REGISTER
001406  000000                  RMCS20: .WORD                   ;CONTROLLER STATUS REGISTER
001410  000000                  RMDS0:  .WORD                   ;DRIVE STATUS REGISTER
001412  000000                  RMER10: .WORD                   ;ERROR REGISTER 1
001414  000000                  RMAS0:  .WORD                   ;ATTENTION SUMMARY REGISTER
001416  000000                  RMLA0:  .WORD                   ;LOOK AHEAD REGISTER
001420  000000                  RMDB0:  .WORD                   ;DATA BUFFER
001422  000000                  RMMR10: .WORD                   ;MAINTENANCE REGISTER #1
001424  000000                  RMDT0:  .WORD                   ;DRIVE TYPE REGISTER
001426  000000                  RMSN0:  .WORD                   ;SERIAL NUMBER REGISTER
001430  000000                  RMOF0:  .WORD                   ;OFFSET REGISTER
001432  000000                  RMDC0:  .WORD                   ;DESIRED CYLINDER REGISTER
001434  000000                  RMCC0:  .WORD                   ;CURRENT CYLINDER REGISTER
001436  000000                  RMMR20: .WORD                   ;MAINTENANCE REGISTER #2
001440  000000                  RMER20: .WORD                   ;ERROR REGISTER 2
001442  000000                  RMEC10: .WORD                   ;ECC POSITION REGISTER
001444  000000                  RMEC20: .WORD                   ;ECC PATTERN REGISTER

                                ;EACH WORD OF THE TEST QUE CONTAINS THE DEVICE NUMBER IN
                                ;THE LOW BYTE AND THE ATTENTION BIT IN THE HIGH BYTE. A ZERO
                                ;WORD IS A BLANK AND REPRESENTS THE END OF THE QUE.

001446  000012                  TSTQUE: .BLKW   10.     ;TEST QUE
```

```
1439                                    ;MEDIA ENABLE IS SET IF THE BAD SECTOR FILES HAVE BEEN RECOVERED
1440                                    ;FOR THE UNIT UNDER TEST, OTHERWISE IT IS ZERO.
1441  001472  000000                    MEDENB: .WORD           ;MEDIA ENABLE
1442
1443                                    ;LOCATIONS "ASNDC" AND "ASNDC" CONTAIN THE CYLINDER, TRACK AND SECTOR
1444                                    ;ADDRESS ASSIGNED BY THE BAD SECTOR MODULE.
1445  001474  000000                    ASNDC:  .WORD           ;ASSIGNED DESIRED CYLINDER
1446  001476  000000                    ASNDA:  .WORD           ;ASSIGNED TRACK, AND SECTOR
1447  001500  000000                    CLKADR: .WORD           ;UNIBUS ADDRESS OF KW11 CLOCK
1448  001502  000000                    CLKVCT: .WORD           ;VECTOR ADDRESS OF KW11 CLOCK
1449
1450                                    ;THE GET INDEX TABLE CONTAINS A BYTE LIST OF REGISTERS WHICH
1451                                    ;ARE READ BY THE GET SUBROUTINE.  THE LIST IS TERMINATED BY
1452                                    ;A NEGATIVE BYTE.
1453  001504  000027                    GETINX: .BLKB   23.             ;GET INDEX TABLE
1454
1455                                    ;THE PUT INDEX TABLE ICONTAINS A BYTE LIST OF REGISTERS WHICH
1456                                    ;ARE WRITTEN BY THE PUT SUBROUTINE.  THE LIST IS TERMINATED BY
1457                                    ;A NEGATIVE BYTE.
1458  001533  000027                    PUTINX: .BLKB   23.             ;PUT INDEX TABLE
1459
1460                                    ;PUT TAGS HERE
1461
```

```
                            .SBTTL  ERROR POINTER TABLE

                            ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                            ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                            ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                            ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                            ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                            ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                            ;*      DH              ;;POINTS TO THE DATA HEADER
                            ;*      DT              ;;POINTS TO THE DATA
                            ;*      DF              ;;POINTS TO THE DATA FORMAT

001562                      $ERRTB:
```

```
1479                                    ;ERROR  1       WRONG UNIT SELECTED
1480  001562  070404                            EMT1
1481  001564  074476                            EHT1
1482  001566  074622                            EDT1
1483  001570  074712                            EFT1
1484
1485
1486                                    ;ERROR  2       DEVICE WENT UNAVAILABLE
1487  001572  070410                            EMT2
1488  001574  074476                            EHT1
1489  001576  074622                            EDT1
1490  001600  074712                            EFT1
1491
1492
1493                                    ;ERROR  3       DEVICE WENT NONEXISTENT
1494  001602  070416                            EMT3
1495  001604  074476                            EHT1
1496  001606  074622                            EDT1
1497  001610  074712                            EFT1
1498
1499
1500                                    ;ERROR  4       CONTROLLER NOT READY
1501  001612  070424                            EMT4
1502  001614  074476                            EHT1
1503  001616  074622                            EDT1
1504  001620  074712                            EFT1
1505
1506
1507                                    ;ERROR  5       DRIVE NOT READY AND GO NOT RESET
1508  001622  070432                            EMT5
1509  001624  074476                            EHT1
1510  001626  074622                            EDT1
1511  001630  074712                            EFT1
1512
1513
1514                                    ;ERROR  6       UNEXPECTED VALUE FOR "ATA" STATUS
1515  001632  070440                            EMT6
1516  001634  074476                            EHT1
1517  001636  074622                            EDT1
1518  001640  074712                            EFT1
1519
1520
1521                                    ;ERROR  7       BUS TIMEOUT TRYING TO READ OR WRITE REGISTER
1522  001642  070446                            EMT7
1523  001644  000000                            0
1524  001646  000000                            0
1525  001650  000000                            0
1526
1527
1528                                    ;ERROR  10      DRIVE NOT READY BUT GO IS RESET
1529  001652  070454                            EMT10
1530  001654  074476                            EHT1
1531  001656  074622                            EDT1
1532  001660  074712                            EFT1
1533
1534
```

```
1535
1536  001662  070460          ;ERROR  11      [illegible]NOT RESET BUT DRIVE IS READY
1537  001664  074476                  EMT11
1538  001666  074622                  EHT1
1539  001670  074712                  EDT1
1540                            EFT1
1541
1542                            ;ERROR  12      INCORRECT FUNCTION CODE
1543  001672  070464                  EMT12
1544  001674  074476                  EHT1
1545  001676  074622                  EDT1
1546  001700  074712                  EFT1
1547
1548
1549                            ;ERROR  13      PARITY ERROR READING REMOTE REGISTERS
1550  001702  070472                  EMT13
1551  001704  074476                  EHT1
1552  001706  074622                  EDT1
1553  001710  074712                  EFT1
1554
1555
1556                            ;ERROR  14      TRANSFER ERROR IS INCORRECT
1557  001712  070504                  EMT14
1558  001714  074476                  EHT1
1559  001716  074622                  EDT1
1560  001720  074712                  EFT1
1561
1562
1563                            ;ERROR  15      INCORRECT WORD COUNT
1564  001722  070512                  EMT15
1565  001724  074476                  EHT1
1566  001726  074622                  EDT1
1567  001730  074712                  EFT1
1568
1569
1570                            ;ERROR  16      INCORRECT BUS ADDRESS
1571  001732  070520                  EMT16
1572  001734  074476                  EHT1
1573  001736  074622                  EDT1
1574  001740  074712                  EFT1
1575
1576
1577                            ;ERROR  17      INCORRECT LBT STATUS
1578  001742  070530                  EMT17
1579  001744  074476                  EHT1
1580  001746  074622                  EDT1
1581  001750  074712                  EFT1
1582
1583
1584                            ;ERROR  20      INCORRECT AOE
1585  001752  070540                  EMT20
1586  001754  074476                  EHT1
1587  001756  074622                  EDT1
1588  001760  074712                  EFT1
1589
1590
```

```
1591
1592  001762  070550          ;ERROR  21      INCORRECT DISK ADDRESS
1593  001764  074476                  EMT21
1594  001766  074622                  EHT1
1595  001770  074712                  EDT1
1596                            EFT1
1597
1598
1599  001772  070560          ;ERROR  22      INCORRECT CYLINDER ADDRESS
1600  001774  074476                  EMT22
1601  001776  074622                  EHT1
1602  002000  074712                  EDT1
1603                            EFT1
1604
1605
1606  002002  070570          ;ERROR  23      INCORRECT WLE STATUS
1607  002004  074476                  EMT23
1608  002006  074622                  EHT1
1609  002010  074712                  EDT1
1610                            EFT1
1611
1612
1613  002012  070600          ;ERROR  24      INCORRECT UPE STATUS
1614  002014  074476                  EMT24
1615  002016  074622                  EHT1
1616  002020  074712                  EDT1
1617                            EFT1
1618
1619
1620  002022  070610          ;ERROR  25      INCORRECT WCF STATUS
1621  002024  074476                  EMT25
1622  002026  074622                  EHT1
1623  002030  074712                  EDT1
1624                            EFT1
1625
1626
1627  002032  070620          ;ERROR  26      INCORRECT WCE STATUS
1628  002034  074476                  EMT26
1629  002036  074622                  EHT1
1630  002040  074712                  EDT1
1631                            EFT1
1632
1633
1634  002042  070630          ;ERROR  27      INCORRECT MDPE STATUS
1635  002044  074476                  EMT27
1636  002046  074622                  EHT1
1637  002050  074712                  EDT1
1638                            EFT1
1639
1640
1641  002052  070640          ;ERROR  30      INCORRECT DCK STATUS
1642  002054  074476                  EMT30
1643  002056  074622                  EHT1
1644  002060  074712                  EDT1
1645                            EFT1
1646
```

```
1647
1648  002062  070650                ;ERROR  31      INCORRECT ECH STATUS
1649  002064  074476                        EMT31
1650  002066  074622                        EHT1
1651  002070  074712                        EDT1
1652                                  EFT1
1653
1654                                  ;ERROR  32      DLT SHOULD NOT BE SET
1655  002072  070660                        EMT32
1656  002074  074476                        EHT1
1657  002076  074622                        EDT1
1658  002100  074712                        EFT1
1659
1660
1661                                  ;ERROR  33      MXF SHOULD NOT BE SET
1662  002102  070670                        EMT33
1663  002104  074476                        EHT1
1664  002106  074622                        EDT1
1665  002110  074712                        EFT1
1666
1667
1668                                  ;ERROR  34      DTE SHOULD NOT BE SET
1669  002112  070700                        EMT34
1670  002114  074476                        EHT1
1671  002116  074622                        EDT1
1672  002120  074712                        EFT1
1673
1674
1675                                  ;ERROR  35      INCORRECT HCRC STATUS
1676  002122  070710                        EMT35
1677  002124  074476                        EHT1
1678  002126  074622                        EDT1
1679  002130  074712                        EFT1
1680
1681
1682                                  ;ERROR  36      INCORRECT HCE STATUS
1683  002132  070720                        EMT36
1684  002134  074476                        EHT1
1685  002136  074622                        EDT1
1686  002140  074712                        EFT1
1687
1688
1689                                  ;ERROR  37      INCORRECT FER STATUS
1690  002142  070730                        EMT37
1691  002144  074476                        EHT1
1692  002146  074622                        EDT1
1693  002150  074712                        EFT1
1694
1695
1696                                  ;ERROR  40      DPE SHOULD NOT BE SET (NOT A DATA COMMAND)
1697  002152  070740                        EMT40
1698  002154  074476                        EHT1
1699  002156  074622                        EDT1
1700  002160  074712                        EFT1
1701
1702
```

```
1703
1704  002162  070746          ;ERROR  41      LOST "MOL" DURING PACK ACKNOWLEDGE
1705  002164  074476                  EMT41
1706  002166  074622                  EHT1
1707  002170  074712                  EDT1
1708                                    EFT1
1709
1710                          ;ERROR  42      UNSAFE ERROR DURING PACK ACKNOWLEDGE
1711  002172  070756                  EMT42
1712  002174  074476                  EHT1
1713  002176  074622                  EDT1
1714  002200  074712                  EFT1
1715
1716
1717                          ;ERROR  43      "OPI" ERROR DURING PACK ACKNOWLEDGE
1718  002202  070770                  EMT43
1719  002204  074476                  EHT1
1720  002206  074622                  EDT1
1721  002210  074712                  EFT1
1722
1723
1724                          ;ERROR  44      "RMR" ERROR DURING PACK ACKNOWLEDGE
1725  002212  071000                  EMT44
1726  002214  074476                  EHT1
1727  002216  074622                  EDT1
1728  002220  074712                  EFT1
1729
1730
1731                          ;ERROR  45      "ILR" ERROR DURING PACK ACKNOWLEDGE
1732  002222  071010                  EMT45
1733  002224  074476                  EHT1
1734  002226  074622                  EDT1
1735  002230  074712                  EFT1
1736
1737
1738                          ;ERROR  46      "ILF" ERROR DURING PACK ACKNOWLEDGE
1739  002232  071020                  EMT46
1740  002234  074476                  EHT1
1741  002236  074622                  EDT1
1742  002240  074712                  EFT1
1743
1744
1745                          ;ERROR  47      COMPOSITE ERROR STATUS IS INCORRECT
1746  002242  071030                  EMT47
1747  002244  074476                  EHT1
1748  002246  074622                  EDT1
1749  002250  074712                  EFT1
1750
1751
1752                          ;ERROR  50      PARITY ERROR WRITING REMOTE REGISTERS
1753  002252  071036                  EMT50
1754  002254  074476                  EHT1
1755  002256  074622                  EDT1
1756  002260  074712                  EFT1
1757
1758
```

```
1759                                    ;ERROR  51      INCORRECT IAE STATUS DURING SEEK COMMAND
1760  002262  071046                              EMT51
1761  002264  074476                              EHT1
1762  002266  074622                              EDT1
1763  002270  074712                              EFT1
1764
1765
1766                                    ;ERROR  52      OPI ERROR DURING SEEK - MEDIUM IS NOT ON LINE
1767  002272  071060                              EMT52
1768  002274  074476                              EHT1
1769  002276  074622                              EDT1
1770  002300  074712                              EFT1
1771
1772
1773                                    ;ERROR  53      OPI ERROR DURING SEEK - MEDIUM IS ON LINE, ASSUME
1774                                    ;               ON CYLINDER LATCH DIDN'T RESET
1775  002302  071076                              EMT53
1776  002304  074476                              EHT1
1777  002306  074622                              EDT1
1778  002310  074712                              EFT1
1779
1780
1781                                    ;ERROR  54      SEEK INCOMPLETE ERROR DURING SEEK COMMAND
1782  002312  071114                              EMT54
1783  002314  074476                              EHT1
1784  002316  074622                              EDT1
1785  002320  074712                              EFT1
1786
1787
1788                                    ;ERROR  55      DEVICE CHECK DURING SEEK COMMAND
1789  002322  071124                              EMT55
1790  002324  074476                              EHT1
1791  002326  074622                              EDT1
1792  002330  074712                              EFT1
1793
1794
1795                                    ;ERROR  56      PIP IS STILL SET AFTER SEEK - SKI IS RESET
1796  002332  071136                              EMT56
1797  002334  074476                              EHT1
1798  002336  074622                              EDT1
1799  002340  074712                              EFT1
1800
1801
1802                                    ;ERROR  57      ATA DID NOT SET DURING SEEK COMMAND
1803  002342  071154                              EMT57
1804  002344  074476                              EHT1
1805  002346  074622                              EDT1
1806  002350  074712                              EFT1
1807
1808
1809                                    ;ERROR  60      IVC ERROR DURING SEEK COMMAND - LOST
1810                                    ;               VOLUME VALID
1811  002352  071164                              EMT60
1812  002354  074476                              EHT1
1813  002356  074622                              EDT1
1814  002360  074712                              EFT1
```

```
1815
1816
1817                                    ;ERROR  61      ERRONEOUS IVC ERROR DURING SEEK COMMAND -
1818                                    ;               VOLUME VALID IS STIL SET
1819  002362  071202                            EMT61
1820  002364  074476                            EHT1
1821  002366  074622                            EDT1
1822  002370  074712                            EFT1
1823
1824
1825                                    ;ERROR  62      MOL IS ZERO, BUT OPI WAS NOT
1826                                    ;               REPORTED DURING SEEK COMMAND
1827  002372  071222                            EMT62
1828  002374  074476                            EHT1
1829  002376  074622                            EDT1
1830  002400  074712                            EFT1
1831
1832
1833                                    ;ERROR  63      UNUSED
1834  002402  000000                            0
1835  002404  000000                            0
1836  002406  000000                            0
1837  002410  000000                            0
1838
1839
1840                                    ;ERROR  64      DRIVE DID NOT DETECT "IVC" ERROR DURING SEEK
1841  002412  071240                            EMT64
1842  002414  074476                            EHT1
1843  002416  074622                            EDT1
1844  002420  074712                            EFT1
1845
1846
1847                                    ;ERROR  65      DRIVE EXECUTED A SEEK WITH ERROR SET
1848  002422  071260                            EMT65
1849  002424  074476                            EHT1
1850  002426  074622                            EDT1
1851  002430  074712                            EFT1
1852
1853
1854                                    ;ERROR  66      UNEXPECTED ERROR SET IN RMER1
1855  002432  071300                            EMT66
1856  002434  074476                            EHT1
1857  002436  074622                            EDT1
1858  002440  074712                            EFT1
1859
1860
1861                                    ;ERROR  67      UNEXPECTED ERROR SET IN RMER2
1862  002442  071312                            EMT67
1863  002444  074476                            EHT1
1864  002446  074622                            EDT1
1865  002450  074712                            EFT1
1866
1867
1868                                    ;ERROR  70      ERRONEOUS "IAE" ERROR DURING RECALIBRATE
1869  002452  071324                            EMT70
1870  002454  074476                            EHT1
```

```
002456  074622                          EDT1
002460  074712                          EFT1


                                ;ERROR  71      "ILF" ERROR DURING RECALIBRATE
002462  071334                          EMT71
002464  074476                          EHT1
002466  074622                          EDT1
002470  074712                          EFT1


                                ;ERROR  72      "OPI" ERROR DURING RECALIBRATE DUE TO "MOL" = 0
002472  071344                          EMT72
002474  074476                          EHT1
002476  074622                          EDT1
002500  074712                          EFT1


                                ;ERROR  73      "OPI" ERROR DURING RECALIBRATE BECAUSE ON
                                ;               CYLINDER DIDNT DROP
002502  071362                          EMT73
002504  074476                          EHT1
002506  074622                          EDT1
002510  074712                          EFT1


                                ;ERROR  74      "IVC" ERROR DURING RECALIBRATE - "VV" = 0
002512  071400                          EMT74
002514  074476                          EHT1
002516  074622                          EDT1
002520  074712                          EFT1


                                ;ERROR  75      ERRONEOUS "IVC" ERROR DURING RECALIBRATE - "VV" = 1
002522  071410                          EMT75
002524  074476                          EHT1
002526  074622                          EDT1
002530  074712                          EFT1


                                ;ERROR  76      "SKI" ERROR DURING RECALIBRATE
002532  071430                          EMT76
002534  074476                          EHT1
002536  074622                          EDT1
002540  074712                          EFT1


                                ;ERROR  77      "DVC" OCCURRED DURING RECALIBRATE
002542  071440                          EMT77
002544  074476                          EHT1
002546  074622                          EDT1
002550  074712                          EFT1


                                ;ERROR  100     LOST "MOL" DURING RECALIBRATE - "OPI" = 0
002552  071452                          EMT100
```

ERROR POINTER TABLE

```
1927  002554  074476                    EHT1
1928  002556  074622                    EDT1
1929  002560  074712                    EFT1
1930
1931
1932                          ;ERROR  101     LOST "VV" DURING RECALIBRATE - "IVC" = 0
1933  002562  071470                    EMT101
1934  002564  074476                    EHT1
1935  002566  074622                    EDT1
1936  002570  074712                    EFT1
1937
1938
1939                          ;ERROR  102     "ATA" DID NOT SET DURING RECALIBRATE
1940  002572  071506                    EMT102
1941  002574  074476                    EHT1
1942  002576  074622                    EDT1
1943  002600  074712                    EFT1
1944
1945
1946                          ;ERROR  103     "OM" DID NOT RESET DURING RECALIBRATE
1947  002602  071516                    EMT103
1948  002604  074476                    EHT1
1949  002606  074622                    EDT1
1950  002610  074712                    EFT1
1951
1952
1953                          ;ERROR  104     "PIP" IS STIL SET AFTER RECALIBRATE
1954  002612  071530                    EMT104
1955  002614  074476                    EHT1
1956  002616  074622                    EDT1
1957  002620  074712                    EFT1
1958
1959
1960                          ;ERROR  105     UNEXPECTED "ILR" ERROR DURING RECALIBRATE
1961  002622  071546                    EMT105
1962  002624  074476                    EHT1
1963  002626  074622                    EDT1
1964  002630  074712                    EFT1
1965
1966
1967                          ;ERROR  106     UNEXPECTED "RMR" ERROR DURING RECALIBRATE
1968  002632  071556                    EMT106
1969  002634  074476                    EHT1
1970  002636  074622                    EDT1
1971  002640  074712                    EFT1
1972
1973
1974                          ;ERROR  107     "UNS" ERROR DURING RECALIBRATE - AC POWER IS LOW
1975  002642  071566                    EMT107
1976  002644  074476                    EHT1
1977  002646  074622                    EDT1
1978  002650  074712                    EFT1
1979
1980
1981                          ;ERROR  110     CANNOT ACCESS MASSBUS CONTROLLER VIA UNIBUS
1982  002652  071606                    EMT110
```

ERROR POINTER TABLE

```
1983  002654  074520                    EHT110
1984  002656  074640                    EDT110
1985  002660  074730                    EFT110
1986
1987
1988                          ;ERROR  111     NONEXISTENT DEVICE
1989  002662  071620                    EMT111
1990  002664  074524                    EHT111
1991  002666  074642                    EDT111
1992  002670  074732                    EFT111
1993
1994
1995                          ;ERROR  112     DEVICE NOT AVAILABLE
1996  002672  071626                    EMT112
1997  002674  074524                    EHT111
1998  002676  074642                    EDT111
1999  002700  074732                    EFT111
2000
2001
2002                          ;ERROR  113     BUS TIMEOUT-NED STATUS FAILURE
2003  002702  071634                    EMT113
2004  002704  000000                    0
2005  002706  000000                    0
2006  002710  000000                    0
2007
2008
2009                          ;ERROR  114     DEVICE NOT AN RM03
2010  002712  071650                    EMT114
2011  002714  074530                    EHT114
2012  002716  074644                    EDT114
2013  002720  074734                    EFT114
2014
2015
2016                          ;ERROR  115     RMCS1 NOT INITIALIZED BY UNIBUS
2017  002722  071656                    EMT115
2018  002724  074476                    EHT1
2019  002726  074622                    EDT1
2020  002730  074712                    EFT1
2021
2022
2023                          ;ERROR  116     RMBA NOT INITIALIZED BY UNIBUS
2024  002732  071666                    EMT116
2025  002734  074476                    EHT1
2026  002736  074622                    EDT1
2027  002740  074712                    EFT1
2028
2029
2030                          ;ERROR  117     RMCS2 NOT INITIALIZED BY UNIBUS
2031  002742  071676                    EMT117
2032  002744  074476                    EHT1
2033  002746  074622                    EDT1
2034  002750  074712                    EFT1
2035
2036
2037                          ;ERROR  120     RMER1 NOT INITIALIZED BY UNIBUS
2038  002752  071706                    EMT120
```

```
002754  074476                          EHT1
002756  074622                          EDT1
002760  074712                          EFT1


                                ;ERROR  121     RMAS NOT INITIALIZED BY UNIBUS
002762  071716                          EMT121
002764  074476                          EHT1
002766  074622                          EDT1
002770  074712                          EFT1


                                ;ERROR  122     RMMR1 NOT INITIALIZED BY UNIBUS
002772  071726                          EMT122
002774  074476                          EHT1
002776  074622                          EDT1
003000  074712                          EFT1


                                ;ERROR  123     RMDS NOT INITIALIZED BY UNIBUS
003002  071736                          EMT123
003004  074476                          EHT1
003006  074622                          EDT1
003010  074712                          EFT1


                                ;ERROR  124     RMEC2 NOT INITIALIZED BY UNIBUS
003012  071746                          EMT124
003014  074476                          EHT1
003016  074622                          EDT1
003020  074712                          EFT1


                                ;ERROR  125     RMMR2 NOT INITIALIZED BY UNIBUS
003022  071756                          EMT125
003024  074476                          EHT1
003026  074622                          EDT1
003030  074712                          EFT1


                                ;ERROR  126     RMCS1 NOT CLEARED BY CONTROLLER CLEAR
003032  071766                          EMT126
003034  074476                          EHT1
003036  074622                          EDT1
003040  074712                          EFT1


                                ;ERROR  127     RMBA NOT CLEARED BY CONTROLLER CLEAR
003042  072000                          EMT127
003044  074476                          EHT1
003046  074622                          EDT1
003050  074712                          EFT1


                                ;ERROR  130     RMCS2 NOT CLEARED BY CONTROLLER CLEAR
003052  072012                          EMT130
```

ERROR POINTER TABLE

```
                                        EHT1
                                        EDT1
                                        EFT1

                        ;ERROR  131     RMER1 NOT CLEARED BY CONTROLLER CLEAR
003062  072024                        EMT131
003064  074476                        EHT1
003066  074622                        EDT1
003070  074712                        EFT1

                        ;ERROR  132     RMAS NOT CLEARED BY CONTROLLER CLEAR
003072  072036                        EMT132
003074  074476                        EHT1
003076  074622                        EDT1
003100  074712                        EFT1

                        ;ERROR  133     RMMR1 NOT CLEARED BY CONTROLLER CLEAR
003102  072050                        EMT133
003104  074476                        EHT1
003106  074622                        EDT1
003110  074712                        EFT1

                        ;ERROR  134     RMDS NOT CLEARED BY CONTROLLER CLEAR
003112  072062                        EMT134
003114  074476                        EHT1
003116  074622                        EDT1
003120  074712                        EFT1

                        ;ERROR  135     RMEC2 NOT CLEARED BY CONTROLLER CLEAR
003122  072074                        EMT135
003124  074476                        EHT1
003126  074622                        EDT1
003130  074712                        EFT1

                        ;ERROR  136     RMMR2 NOT CLEARED BY CONTROLLER CLEAR
003132  072106                        EMT136
003134  074476                        EHT1
003136  074622                        EDT1
003140  074712                        EFT1

                        ;ERROR  137     RMCS1 NOT CLEARED BY ERROR CLEAR
003142  072120                        EMT137
003144  074476                        EHT1
003146  074622                        EDT1
003150  074712                        EFT1

                        ;ERROR  140     RMCS2 NOT CLEARED BY ERROR CLEAR
003152  072130                        EMT140
```

```
003154  074476                          EHT1
003156  074622                          EDT1
003160  074712                          EFT1


                                ;ERROR  141     RMCS1 NOT CLEARED BY DRIVE CLEAR
003162  072140                          EMT141
003164  074476                          EHT1
003166  074622                          EDT1
003170  074712                          EFT1


                                ;ERROR  142     RMDS NOT CLEARED BY DRIVE CLEAR
003172  072150                          EMT142
003174  074476                          EHT1
003176  074622                          EDT1
003200  074712                          EFT1


                                ;ERROR  143     RMER1 NOT CLEARED BY DRIVE CLEAR
003202  072160                          EMT143
003204  074476                          EHT1
003206  074622                          EDT1
003210  074712                          EFT1


                                ;ERROR  144     RMAS NOT CLEARED BY DRIVE CLEAR
003212  072170                          EMT144
003214  074476                          EHT1
003216  074622                          EDT1
003220  074712                          EFT1


                                ;ERROR  145     RMMR1  NOT CLEARED BY DRIVE CLEAR
003222  072200                          EMT145
003224  074476                          EHT1
003226  074622                          EDT1
003230  074712                          EFT1


                                ;ERROR  146     RMMR2 NOT CLEARED BY DRIVE CLEAR
003232  072210                          EMT146
003234  074476                          EHT1
003236  074622                          EDT1
003240  074712                          EFT1


                                ;ERROR  147     RMER2 NOT CLEARED BY DRIVE CLEAR
003242  072220                          EMT147
003244  074476                          EHT1
003246  074622                          EDT1
003250  074712                          EFT1


                                ;ERROR  150     RMEC2 NOT CLEARED BY DRIVE CLEAR
003252  072230                          EMT150
```

```
 2207  003254  074476                    EHT1
 2208  003256  074622                    EDT1
 2209  003260  074712                    EFT1
 2210
 2211
 2212                              ;ERROR  151     MEDIUM NOT ON LINE
 2213  003262  072240                    EMT151
 2214  003264  074476                    EHT1
 2215  003266  074622                    EDT1
 2216  003270  074712                    EFT1
 2217
 2218
 2219                              ;ERROR  152     DRIVE FAULT
 2220  003272  072252                    EMT152
 2221  003274  074476                    EHT1
 2222  003276  074622                    EDT1
 2223  003300  074712                    EFT1
 2224
 2225
 2226                              ;ERROR  153     UNSAFE SHOULD BE SET BECAUSE DVC IS SET
 2227  003302  072264                    EMT153
 2228  003304  074476                    EHT1
 2229  003306  074622                    EDT1
 2230  003310  074712                    EFT1
 2231
 2232
 2233                              ;ERROR  154     UNSAFE SHOULD NOT BE SET, AC IS LOW
 2234  003312  072302                    EMT154
 2235  003314  074476                    EHT1
 2236  003316  074622                    EDT1
 2237  003320  074712                    EFT1
 2238
 2239
 2240                              ;ERROR  155     VOLUME VALID NOT SET BY PACK ACK
 2241  003322  072320                    EMT155
 2242  003324  074476                    EHT1
 2243  003326  074622                    EDT1
 2244  003330  074712                    EFT1
 2245
 2246
 2247                              ;ERROR  156     OFFSET MODE NOT SET BY OFFSET COMMAND
 2248  003332  072332                    EMT156
 2249  003334  074476                    EHT1
 2250  003336  074622                    EDT1
 2251  003340  074712                    EFT1
 2252
 2253
 2254                              ;ERROR  157     OFFSET MODE NOT RESET BY RTC COMMAND
 2255  003342  072344                    EMT157
 2256  003344  074476                    EHT1
 2257  003346  074622                    EDT1
 2258  003350  074712                    EFT1
 2259
 2260
 2261                              ;ERROR  160     RMOF NOT RESET BY RIP COMMAND
 2262  003352  072356                    EMT160
```

```
2263  003354  074476                    EHT1
2264  003356  074622                    EDT1
2265  003360  074712                    EFT1
2266
2267
2268                              ;ERROR  161     RMDA NOT RESET BY RIP COMMAND
2269  003362  072366                    EMT161
2270  003364  074476                    EHT1
2271  003366  074622                    EDT1
2272  003370  074712                    EFT1
2273
2274
2275                              ;ERROR  162     RMDC NOT RESET BY RIP COMMAND
2276  003372  072400                    EMT162
2277  003374  074476                    EHT1
2278  003376  074622                    EDT1
2279  003400  074712                    EFT1
2280
2281
2282                              ;ERROR 163      DATA WAS ECC CORRECTED BUT DOES NOT COMPARE WITH
2283                              ;               WRITE BUFFER
2284  003402  074272                    EMT336
2285  003404  074560                    EHT336
2286  003406  074656                    EDT336
2287  003410  074746                    EFT336
2288
2289
2290                              ;ERROR  164     OPI SHOULD NOT BE SET
2291  003412  072422                    EMT164
2292  003414  074476                    EHT1
2293  003416  074622                    EDT1
2294  003420  074712                    EFT1
2295
2296
2297                              ;ERROR  165     IVC SHOULD NOT BE SET
2298  003422  072430                    EMT165
2299  003424  074476                    EHT1
2300  003426  074622                    EDT1
2301  003430  074712                    EFT1
2302
2303
2304                              ;ERROR  166     IAE SHOULD NOT BE SET
2305  003432  072436                    EMT166
2306  003434  074476                    EHT1
2307  003436  074622                    EDT1
2308  003440  074712                    EFT1
2309
2310
2311                              ;ERROR  167     NEM SHOULD NOT BE SET
2312  003442  072444                    EMT167
2313  003444  074476                    EHT1
2314  003446  074622                    EDT1
2315  003450  074712                    EFT1
2316
2317
2318                              ;ERROR 170      UNUSED
```

```
2319  003452  000000                        0
2320  003454  000000                        0
2321  003456  000000                        0
2322  003460  000000                        0
2323
2324
2325                                ;ERROR  171     "ATA" NOT SET DURING RETURN TO CENTERLINE
2326  003462  072462                        EMT171
2327  003464  074476                        EHT1
2328  003466  074622                        EDT1
2329  003470  074712                        EFT1
2330
2331
2332                                ;ERROR  172     "ATA" NOT SET BY OFFSET COMMAND
2333  003472  072472                        EMT172
2334  003474  074476                        EHT1
2335  003476  074622                        EDT1
2336  003500  074712                        EFT1
2337
2338
2339                                ;ERROR  173     RMER2 NOT INITIALIZED BY UNIBUS INIT
2340  003502  072502                        EMT173
2341  003504  074476                        EHT1
2342  003506  074622                        EDT1
2343  003510  074712                        EFT1
2344
2345
2346                                ;ERROR  174     RMER2 NOT INITIALIZED BY CONTROLLER CLEAR
2347  003512  072512                        EMT174
2348  003514  074476                        EHT1
2349  003516  074622                        EDT1
2350  003520  074712                        EFT1
2351
2352
2353                                ;ERROR  175     SELECTED DEVICE IS IN WRITE PROTECT
2354  003522  072524                        EMT175
2355  003524  074476                        EHT1
2356  003526  074622                        EDT1
2357  003530  074712                        EFT1
2358
2359
2360                                ;ERROR  176     CANNOT SET DIAGNOSTIC MODE
2361  003532  072532                        EMT176
2362  003534  074476                        EHT1
2363  003536  074622                        EDT1
2364  003540  074712                        EFT1
2365
2366
2367                                ;ERROR  177     INCORRECT "MOL" STATUS DURING DIAGNOSTIC MODE
2368  003542  072540                        EMT177
2369  003544  074476                        EHT1
2370  003546  074622                        EDT1
2371  003550  074712                        EFT1
2372
2373
2374                                ;ERROR  200     INCORRECT "PIP" STATUS DURING DIAGNOSTIC MODE
```

ERROR POINTER TABLE

```
2375  003552  072552                    EMT200
2376  003554  074476                    EHT1
2377  003556  074622                    EDT1
2378  003560  074712                    EFT1
2379
2380
2381                              ;ERROR  201     INCORRECT "WRL" STATUS DURING DIAGNOSTIC MODE
2382  003562  072564                    EMT201
2383  003564  074476                    EHT1
2384  003566  074622                    EDT1
2385  003570  074712                    EFT1
2386
2387
2388                              ;ERROR  202     INCORRECT "SKI" STATUS DURING DIAGNOSTIC MODE
2389  003572  072576                    EMT202
2390  003574  074476                    EHT1
2391  003576  074622                    EDT1
2392  003600  074712                    EFT1
2393
2394
2395                              ;ERROR  203     INCORRECT "DVC" STATUS DURING DIAGNOSTIC MODE
2396  003602  072610                    EMT203
2397  003604  074476                    EHT1
2398  003606  074622                    EDT1
2399  003610  074712                    EFT1
2400
2401
2402                              ;ERROR  204     "VV" WAS NOT RESET BY MAINTENANCE UNIT READY
2403  003612  072622                    EMT204
2404  003614  074476                    EHT1
2405  003616  074622                    EDT1
2406  003620  074712                    EFT1
2407
2408
2409                              ;ERROR  205     SELECTED DEVICE HAS A PERSISTENT "SKI" ERROR
2410  003622  072640                    EMT205
2411  003624  074476                    EHT1
2412  003626  074622                    EDT1
2413  003630  074712                    EFT1
2414
2415
2416                              ;ERROR  206     "LBC" DID NOT SET DURING DIAGNOSTIC MODE
2417  003632  072650                    EMT206
2418  003634  074476                    EHT1
2419  003636  074622                    EDT1
2420  003640  074712                    EFT1
2421
2422
2423                              ;ERROR  207     UNEXPECTED LOSS OF "MOL" - MEDIUM IS OFF LINE
2424  003642  072660                    EMT207
2425  003644  074476                    EHT1
2426  003646  074622                    EDT1
2427  003650  074712                    EFT1
2428
2429
2430                              ;ERROR  210     UNEXPECTED LOSS OF VOLUME VALID - "VV" = 0
```

```
2431  003652  072672                    EMT210
2432  003654  074476                    EHT1
2433  003656  074622                    EDT1
2434  003660  074712                    EFT1
2435
2436
2437                          ;ERROR  211     UNEXPECTED MECHANICAL MOTION - "PIP" = 1
2438  003662  072700                    EMT211
2439  003664  074476                    EHT1
2440  003666  074622                    EDT1
2441  003670  074712                    EFT1
2442
2443
2444                          ;ERROR  212     UNEXPECTED DEVICE FAULT - "DVC" = 1
2445  003672  072714                    EMT212
2446  003674  074476                    EHT1
2447  003676  074622                    EDT1
2448  003700  074712                    EFT1
2449
2450
2451                          ;ERROR  213     UNEXPECTED SEEK INCOMPLETE ERROR - "SKI" = 1
2452  003702  072730                    EMT213
2453  003704  074476                    EHT1
2454  003706  074622                    EDT1
2455  003710  074712                    EFT1
2456
2457
2458                          ;ERROR  214     DRIVE EXECUTED A RECALIBRATE WITH ERROR SET
2459  003712  072740                    EMT214
2460  003714  074510                    EHT2
2461  003716  074632                    EDT2
2462  003720  074722                    EFT2
2463
2464
2465                          ;ERROR  215     DRIVE DID NOT DETECT "IVC" ERROR DURING RECALIBRATE
2466  003722  072760                    EMT215
2467  003724  074510                    EHT2
2468  003726  074632                    EDT2
2469  003730  074722                    EFT2
2470
2471
2472                          ;ERROR  216     INCORRECT "IVC" STATUS
2473  003732  072772                    EMT216
2474  003734  074476                    EHT1
2475  003736  074622                    EDT1
2476  003740  074712                    EFT1
2477
2478
2479                          ;ERROR  217     INCORRECT "IAE" STATUS
2480  003742  073002                    EMT217
2481  003744  074476                    EHT1
2482  003746  074622                    EDT1
2483  003750  074712                    EFT1
2484
2485
2486                          ;ERROR 220      INCORRECT "WLE" STATUS
```

ERROR POINTER TABLE

```
2487  003752  073012                    EMT220
2488  003754  074476                    EHT1
2489  003756  074622                    EDT1
2490  003760  074712                    EFT1
2491
2492
2493                              ;ERROR  221     INCORRECT "OPI" STATUS
2494  003762  073022                    EMT221
2495  003764  074476                    EHT1
2496  003766  074622                    EDT1
2497  003770  074712                    EFT1
2498
2499
2500                              ;ERROR  222     RM03 DID NOT DETECT RMR ERROR
2501  003772  073032                    EMT222
2502  003774  074476                    EHT1
2503  003776  074622                    EDT1
2504  004000  074712                    EFT1
2505
2506
2507                              ;ERROR  223     RM03 DID NOT DETECT PARITY ERROR ON MASSBUS CONTROL BUS
2508  004002  073042                    EMT223
2509  004004  074534                    EHT223
2510  004006  074646                    EDT223
2511  004010  074736                    EFT223
2512
2513
2514                              ;ERROR  224     UNUSED
2515  004012  000000                    0
2516  004014  000000                    0
2517  004016  000000                    0
2518  004020  000000                    0
2519
2520
2521                              ;ERROR  225     UNUSED
2522  004022  000000                    0
2523  004024  000000                    0
2524  004026  000000                    0
2525  004030  000000                    0
2526
2527
2528                              ;ERROR  226     UNUSED
2529  004032  000000                    0
2530  004034  000000                    0
2531  004036  000000                    0
2532  004040  000000                    0
2533
2534
2535                              ;ERROR  227     UNUSED
2536  004042  000000                    0
2537  004044  000000                    0
2538  004046  000000                    0
2539  004050  000000                    0
2540
2541
2542                              ;ERROR  230     UNUSED
```

```
2543  004052  000000                    0
2544  004054  000000                    0
2545  004056  000000                    0
2546  004060  000000                    0
2547
2548
2549                          ;ERROR  231     UNUSED
2550  004062  000000                    0
2551  004064  000000                    0
2552  004066  000000                    0
2553  004070  000000                    0
2554
2555
2556                          ;ERROR  232     UNUSED
2557  004072  000000                    0
2558  004074  000000                    0
2559  004076  000000                    0
2560  004100  000000                    0
2561
2562
2563                          ;ERROR  233     UNUSED
2564  004102  000000                    0
2565  004104  000000                    0
2566  004106  000000                    0
2567  004110  000000                    0
2568
2569
2570                          ;ERROR  234     UNUSED
2571  004112  000000                    0
2572  004114  000000                    0
2573  004116  000000                    0
2574  004120  000000                    0
2575
2576
2577                          ;ERROR  235     UNUSED
2578  004122  000000                    0
2579  004124  000000                    0
2580  004126  000000                    0
2581  004130  000000                    0
2582
2583
2584                          ;ERROR  236     UNUSED
2585  004132  000000                    0
2586  004134  000000                    0
2587  004136  000000                    0
2588  004140  000000                    0
2589
2590
2591                          ;ERROR  237     UNUSED
2592  004142  000000                    0
2593  004144  000000                    0
2594  004146  000000                    0
2595  004150  000000                    0
2596
2597
2598                          ;ERROR  240     UNUSED
```

```
2599  004152  000000                    0
2600  004154  000000                    0
2601  004156  000000                    0
2602  004160  000000                    0
2603
2604
2605                          ;ERROR  241     UNUSED
2606  004162  000000                    0
2607  004164  000000                    0
2608  004166  000000                    0
2609  004170  000000                    0
2610
2611
2612                          ;ERROR  242     UNUSED
2613  004172  000000                    0
2614  004174  000000                    0
2615  004176  000000                    0
2616  004200  000000                    0
2617
2618
2619                          ;ERROR  243     UNUSED
2620  004202  000000                    0
2621  004204  000000                    0
2622  004206  000000                    0
2623  004210  000000                    0
2624
2625
2626                          ;ERROR  244     UNUSED
2627  004212  000000                    0
2628  004214  000000                    0
2629  004216  000000                    0
2630  004220  000000                    0
2631
2632
2633                          ;ERROR  245     UNUSED
2634  004222  000000                    0
2635  004224  000000                    0
2636  004226  000000                    0
2637  004230  000000                    0
2638
2639
2640                          ;ERROR  246     "ATA" NOT RESET BY GO WHEN "ERR" = 0
2641  004232  073116                    EMT246
2642  004234  074476                    EHT1
2643  004236  074622                    EDT1
2644  004240  074712                    EFT1
2645
2646
2647                          ;ERROR  247     "ATA" NOT RESET BY WRITING RMAS
2648  004242  073126                    EMT247
2649  004244  074476                    EHT1
2650  004246  074622                    EDT1
2651  004250  074712                    EFT1
2652
2653
2654                          ;ERROR  250     "ATA" WAS RESET BY GO WHEN "ERR" = 1
```

```
2655  004252  073140                        EMT250
2656  004254  074476                        EHT1
2657  004256  074622                        EDT1
2658  004260  074712                        EFT1
2659
2660
2661                          ;ERROR  251     PROGRAM INTERRUPT WAS NOT GENERATED
2662  004262  073154                        EMT251
2663  004264  074510                        EHT2
2664  004266  074632                        EDT2
2665  004270  074722                        EFT2
2666
2667
2668                          ;ERROR  252     PROGRAM INTERRUPT SHOULD NOT HAVE BEEN GENERATED
2669  004272  073162                        EMT252
2670  004274  074510                        EHT2
2671  004276  074632                        EDT2
2672  004300  074722                        EFT2
2673
2674
2675                          ;ERROR  253     OFFSET MODE WAS NOT RESET BY WRITING RMDC
2676  004302  073170                        EMT253
2677  004304  074476                        EHT1
2678  004306  074622                        EDT1
2679  004310  074712                        EFT1
2680
2681
2682                          ;ERROR  254     INCORRECT "ILF" STATUS
2683  004312  073206                        EMT254
2684  004314  074476                        EHT1
2685  004316  074622                        EDT1
2686  004320  074712                        EFT1
2687
2688
2689                          ;ERROR  255     INCORRECT "ATA" STATUS
2690  004322  073216                        EMT255
2691  004324  074476                        EHT1
2692  004326  074622                        EDT1
2693  004330  074712                        EFT1
2694
2695
2696                          ;ERROR  256     INCORRECT "ILR" STATUS
2697  004332  073226                        EMT256
2698  004334  074546                        EHT256
2699  004336  074646                        EDT223
2700  004340  074736                        EFT223
2701
2702
2703                          ;ERROR  257     INVALID IAE STATUS DURING SEARCH COMMAND
2704  004342  073236                        EMT257
2705  004344  074476                        EHT1
2706  004346  074622                        EDT1
2707  004350  074712                        EFT1
2708
2709
2710                          ;ERROR  260     "IVC" WAS NOT DETECTED DURING SEARCH COMMAND
```

```
2711  004352  073250                    EMT260
2712  004354  074476                    EHT1
2713  004356  074622                    EDT1
2714  004360  074712                    EFT1
2715
2716
2717                          ;ERROR  261     DRIVE EXECUTED SEARCH WITH ERROR SET
2718  004362  073262                    EMT261
2719  004364  074476                    EHT1
2720  004366  074622                    EDT1
2721  004370  074712                    EFT1
2722
2723
2724                          ;ERROR  262     "LBC" ERROR NOT SET DURING DIAGNOSTIC MODE
2725  004372  073302                    EMT262
2726  004374  074476                    EHT1
2727  004376  074622                    EDT1
2728  004400  074712                    EFT1
2729
2730
2731                          ;ERROR  263     "SKI" ERROR DURING SEARCH COMMAND
2732  004402  073312                    EMT263
2733  004404  074476                    EHT1
2734  004406  074622                    EDT1
2735  004410  074712                    EFT1
2736
2737
2738                          ;ERROR  264     "IVC" ERROR DURING SEARCH - LOST VOLUME VALID
2739  004412  073322                    EMT264
2740  004414  074476                    EHT1
2741  004416  074622                    EDT1
2742  004420  074712                    EFT1
2743
2744
2745                          ;ERROR  265     ERRONEOUS IVC ERROR DURING SEARCH - VOLUME IS VALID
2746  004422  073342                    EMT265
2747  004424  074476                    EHT1
2748  004426  074622                    EDT1
2749  004430  074712                    EFT1
2750
2751
2752                          ;ERROR  266     DEVICE FAULT (DVC) DURING SEARCH
2753  004432  073362                    EMT266
2754  004434  074476                    EHT1
2755  004436  074622                    EDT1
2756  004440  074712                    EFT1
2757
2758
2759                          ;ERROR  267     SKI SHOULD HAVE BEEN SET BECAUSE CYLINDER
2760                          ;               ADDRESS IS TOO LARGE
2761  004442  073374                    EMT267
2762  004444  074476                    EHT1
2763  004446  074622                    EDT1
2764  004450  074712                    EFT1
2765
2766
```

```
                                ;ERROR  270     OPI ERROR DURING SEARCH BECAUSE MOL = 0
004452  073412                          EMT270
004454  074476                          EHT1
004456  074622                          EDT1
004460  074712                          EFT1


                                ;ERROR  271     OPI ERROR DURING SEARCH BECAUSE ON CYLINDER
                                ;               DIDN'T DROP
004462  073426                          EMT271
004464  074476                          EHT1
004466  074622                          EDT1
004470  074712                          EFT1


                                ;ERROR  272     LOST MOL DURING SEARCH, OPI IS NOT SET
004472  073444                          EMT272
004474  074476                          EHT1
004476  074622                          EDT1
004500  074712                          EFT1


                                ;ERROR  273     PIP STIL SET AFTER SEARCH
004502  073462                          EMT273
004504  074476                          EHT1
004506  074622                          EDT1
004510  074712                          EFT1


                                ;ERROR  274     PARITY ERROR OCCURRED WHILE WRITING REMOTE
                                ;               REGISTERS BUT MXF DID NOT SET
004512  073500                          EMT274
004514  074476                          EHT1
004516  074622                          EDT1
004520  074712                          EFT1


                                ;ERROR  275     MXF ERROR - COMPOSITE ERROR OCCURRED BEFORE DATA
                                ;               COMMAND STARTED
004522  073516                          EMT275
004524  074476                          EHT1
004526  074622                          EDT1
004530  074712                          EFT1


                                ;ERROR  276     "OPI" ERROR DURING DATA TRANSFER BECAUSE "MOL" WAS
                                ;               ZERO
004532  073530                          EMT276
004534  074476                          EHT1
004536  074622                          EDT1
004540  074712                          EFT1


                                ;ERROR  277     "OPI" ERROR DURING DATA TRANSFER BECAUSE 1) ON
                                ;               CYLINDER DIDN'T DROP OR 2) SEARCH TIMED OUT OR
                                ;               3) RUN TIMED OUT
```

ERROR POINTER TABLE

```
2823  004542  073544                        EMT277
2824  004544  074476                        EHT1
2825  004546  074622                        EDT1
2826  004550  074712                        EFT1
2827
2828
2829                          ;ERROR  300     "IVC" ERROR DURING DATA TRANSFER BECAUSE VOLUME .
2830                          ;               WAS NOT VALID
2831  004552  073562                        EMT300
2832  004554  074476                        EHT1
2833  004556  074622                        EDT1
2834  004560  074712                        EFT1
2835
2836
2837                          ;ERROR  301     ERRONEOUS "IVC" ERROR DURING DATA TRANSFER - VOLUME
2838                          ;               IS VALID
2839  004562  073602                        EMT301
2840  004564  074476                        EHT1
2841  004566  074622                        EDT1
2842  004570  074712                        EFT1
2843
2844
2845                          ;ERROR  302     "ILR" ERROR DURING DATA TRANSFER
2846  004572  073624                        EMT302
2847  004574  074476                        EHT1
2848  004576  074622                        EDT1
2849  004600  074712                        EFT1
2850
2851
2852                          ;ERROR  303     "ILF" ERROR DURING DATA TRANSFER
2853  004602  073636                        EMT303
2854  004604  074476                        EHT1
2855  004606  074622                        EDT1
2856  004610  074712                        EFT1
2857
2858
2859                          ;ERROR  304     "RMR" ERROR DURING DATA TRANSFER
2860  004612  073650                        EMT304
2861  004614  074476                        EHT1
2862  004616  074622                        EDT1
2863  004620  074712                        EFT1
2864
2865
2866                          ;ERROR  305     INCORRECT "IAE" STATUS DURING DATA TRANSFER
2867  004622  073662                        EMT305
2868  004624  074476                        EHT1
2869  004626  074622                        EDT1
2870  004630  074712                        EFT1
2871
2872
2873                          ;ERROR  306     "SKI" ERROR DURING DATA TRANSFER
2874  004632  073674                        EMT306
2875  004634  074476                        EHT1
2876  004636  074622                        EDT1
2877  004640  074712                        EFT1
2878
```

```
2879
2880                                    ;ERROR  307     DRIVE DID NOT DETECT SKI ERROR DUE TO CYLINDER
2881  004642  073704                              EMT307
2882  004644  074476                              EHT1
2883  004646  074622                              EDT1
2884  004650  074712                              EFT1
2885
2886
2887                                    ;ERROR  310     DEVICE FAULT DURING DATA TRANSFER
2888  004652  073724                              EMT310
2889  004654  074476                              EHT1
2890  004656  074622                              EDT1
2891  004660  074712                              EFT1
2892
2893
2894                                    ;ERROR  311     LOSS OF BIT CLOCK DURING DATA TRANSFER
2895  004662  073736                              EMT311
2896  004664  074476                              EHT1
2897  004666  074622                              EDT1
2898  004670  074712                              EFT1
2899
2900
2901                                    ;ERROR  312     LOSS OF SYSTEM CLOCK DURING DATA TRANSFER
2902  004672  073750                              EMT312
2903  004674  074476                              EHT1
2904  004676  074622                              EDT1
2905  004700  074712                              EFT1
2906
2907
2908                                    ;ERROR  313     UNSAFE ERROR DURING DATA TRANSFER (DVC = 0)
2909  004702  073762                              EMT313
2910  004704  074476                              EHT1
2911  004706  074622                              EDT1
2912  004710  074712                              EFT1
2913
2914
2915                                    ;ERROR  314     DRIVE TIMING ERROR DURING DATA TRANSFER
2916  004712  074002                              EMT314
2917  004714  074476                              EHT1
2918  004716  074622                              EDT1
2919  004720  074712                              EFT1
2920
2921
2922                                    ;ERROR  315     WRITE LOCK ERROR
2923  004722  074014                              EMT315
2924  004724  074476                              EHT1
2925  004726  074622                              EDT1
2926  004730  074712                              EFT1
2927
2928
2929                                    ;ERROR  316     ERRONEOUS WRITE LOCK ERROR
2930  004732  074026                              EMT316
2931  004734  074476                              EHT1
2932  004736  074622                              EDT1
2933  004740  074712                              EFT1
2934
```

```
2935
2936                                    ;ERROR  317     HEADER CRC ERROR DURING DATA TRANSFER
2937  004742  074040                              EMT317
2938  004744  074476                              EHT1
2939  004746  074622                              EDT1
2940  004750  074712                              EFT1
2941
2942
2943                                    ;ERROR  320     FORMAT ERROR DURING DATA TRANSFER
2944  004752  074050                              EMT320
2945  004754  074476                              EHT1
2946  004756  074622                              EDT1
2947  004760  074712                              EFT1
2948
2949
2950                                    ;ERROR  321     HEADER COMPARE ERROR DURING DATA TRANSFER
2951  004762  074060                              EMT321
2952  004764  074476                              EHT1
2953  004766  074622                              EDT1
2954  004770  074712                              EFT1
2955
2956
2957                                    ;ERROR  322     HEADER ERRORS SHOULD NOT BE SET
2958  004772  074070                              EMT322
2959  004774  074476                              EHT1
2960  004776  074622                              EDT1
2961  005000  074712                              EFT1
2962
2963
2964                                    ;ERROR  323     DATA CHECK ERROR DURING DATA TRANSFER
2965  005002  074076                              EMT323
2966  005004  074476                              EHT1
2967  005006  074622                              EDT1
2968  005010  074712                              EFT1
2969
2970
2971                                    ;ERROR  324     CORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
2972  005012  074106                              EMT324
2973  005014  074476                              EHT1
2974  005016  074622                              EDT1
2975  005020  074712                              EFT1
2976
2977
2978                                    ;ERROR  325     UNCORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
2979  005022  074120                              EMT325
2980  005024  074476                              EHT1
2981  005026  074622                              EDT1
2982  005030  074712                              EFT1
2983
2984
2985                                    ;ERROR  326     DATA PARITY ERROR DURING READ COMMMAND
2986  005032  074132                              EMT326
2987  005034  074476                              EHT1
2988  005036  074622                              EDT1
2989  005040  074712                              EFT1
2990
```

```
2991
2992
2993  005042  074150          ;ERROR  327     OFFSET MODE NOT RESET BY WRITE COMMAND
2994  005044  074476                  EMT327
2995  005046  074622                  EHT1
2996  005050  074712                  EDT1
2997                            EFT1
2998
2999                            ;ERROR  330     DATA PARITY ERROR DURING WRITE COMMAND
3000  005052  074162                  EMT330
3001  005054  074476                  EHT1
3002  005056  074622                  EDT1
3003  005060  074712                  EFT1
3004
3005
3006                            ;ERROR  331     WRITE CLOCK FALURE DURING WRITE COMMAND
3007  005062  074172                  EMT331
3008  005064  074476                  EHT1
3009  005066  074622                  EDT1
3010  005070  074712                  EFT1
3011
3012
3013                            ;ERROR  332     DATA LATE ERROR DURING DATA TRANSFER
3014  005072  074204                  EMT332
3015  005074  074476                  EHT1
3016  005076  074622                  EDT1
3017  005100  074712                  EFT1
3018
3019
3020                            ;ERROR  333     PIP STIL SET AFTER DATA TRANSFER - SKI = 0
3021  005102  074216                  EMT333
3022  005104  074476                  EHT1
3023  005106  074622                  EDT1
3024  005110  074712                  EFT1
3025
3026
3027                            ;ERROR  334     LOST MOL DURING DATA TRANSFER - OPI = 0
3028  005112  074234                  EMT334
3029  005114  074476                  EHT1
3030  005116  074622                  EDT1
3031  005120  074712                  EFT1
3032
3033
3034                            ;ERROR  335     LOST VOLUME VALID DURING DATA TRANSFER - IVC  = 0
3035  005122  074252                  EMT335
3036  005124  074476                  EHT1
3037  005126  074622                  EDT1
3038  005130  074712                  EFT1
3039
3040
3041                            ;ERROR  336     DATA READ DOES NOT COMPARE WITH DATA WRITTEN
3042  005132  074272                  EMT336
3043  005134  074560                  EHT336
3044  005136  074656                  EDT336
3045  005140  074746                  EFT336
3046
```

```
3047
3048                                    ;ERROR  337     WRITE CHECK ERROR NOT DETECTED
3049  005142  074302                              EMT337
3050  005144  074572                              EHT337
3051  005146  074666                              EDT337
3052  005150  074756                              EFT337
3053
3054
3055                                    ;ERROR  340     WRITE CHECK ERROR AT UNEXPECTED ADDRESS
3056  005152  074312                              EMT340
3057  005154  074560                              EHT336
3058  005156  074656                              EDT336
3059  005160  074746                              EFT336
3060
3061
3062                                    ;ERROR  341     INCORRECT DATA DURING WRITE CHECK ERROR
3063  005162  074324                              EMT341
3064  005164  074560                              EHT336
3065  005166  074656                              EDT336
3066  005170  074746                              EFT336
3067
3068
3069                                    ;ERROR  342     "IVC" ERROR NOT DETECTED DURING DATA TRANSFER
3070  005172  074332                              EMT342
3071  005174  074476                              EHT1
3072  005176  074622                              EDT1
3073  005200  074712                              EFT1
3074
3075
3076                                    ;ERROR  343     "FER" NOT DETECTED DURING DATA TRANSFER
3077  005202  074344                              EMT343
3078  005204  074476                              EHT1
3079  005206  074622                              EDT1
3080  005210  074712                              EFT1
3081
3082
3083                                    ;ERROR  344     "HCE" NOT DETECTED DURING DATA TRANSFER
3084  005212  074356                              EMT344
3085  005214  074604                              EHT344
3086  005216  074676                              EDT344
3087  005220  074766                              EFT344
3088
3089
3090                                    ;ERROR  345     "BSE" NOT DETECTED DURING DATA TRANSFER
3091  005222  074370                              EMT345
3092  005224  074476                              EHT1
3093  005226  074622                              EDT1
3094  005230  074712                              EFT1
3095
3096
3097                                    ;ERROR  346     HEADER ERROR WAS DETECTED W/ HCI SET
3098  005232  074400                              EMT346
3099  005234  074476                              EHT1
3100  005236  074622                              EDT1
3101  005240  074712                              EFT1
3102
```

```
3103
3104                                    ;ERROR  347     DATA TRANSFER NOT ABORTED W/ COMP ERROR SET
3105  005242  074414                              EMT347
3106  005244  074476                              EHT1
3107  005246  074622                              EDT1
3108  005250  074712                              EFT1
3109
3110
3111                                    ;ERROR  350     LOST VOLUME VALID DURING SEARCH - "IVC" = 0
3112  005252  074426                              EMT350
3113  005254  074476                              EHT1
3114  005256  074622                              EDT1
3115  005260  074712                              EFT1
3116
3117
3118                                    ;ERROR  351     "ATA" DID NOT SET DURING SEARCH
3119  005262  074444                              EMT351
3120  005264  074476                              EHT1
3121  005266  074622                              EDT1
3122  005270  074712                              EFT1
3123
3124
3125                                    ;ERROR  352     PROGRAM TIMEOUT WHILE TESTING RMLA
3126  005272  074454                              EMT352
3127  005274  000000                              0
3128  005276  000000                              0
3129  005300  000000                              0
3130
3131
3132                                    ;ERROR  353     LOOK AHEAD TEST FAILS
3133  005302  074460                              EMT353
3134  005304  074616                              EHT353
3135  005306  074710                              EDT353
3136  005310  074776                              EFT353
3137
3138
3139                                    ;ERROR  354     BSE SHOULD NOT BE SET
3140  005312  074470                              EMT354
3141  005314  074476                              EHT1
3142  005316  074622                              EDT1
3143  005320  074712                              EFT1
3144
3145
3146                                    ;PUT ERROR TABLE HERE
3147                                    .SBTTL  ERROR TABLE USAGE
3148
3149                                    ;THE ERROR TABLE ABOVE CONSISTS OF FOUR WORD ENTRIES FOR EACH ERROR
3150                                    ;NUMBER, I.E.,
3151                                    ;
3152                                    ;       EMT - ERROR MESSAGE TABLE ADDRESS
3153                                    ;       EHT - ERROR HEADER TABLE ADDRESS
3154                                    ;       EDT - ERROR DATA TABLE ADDRESS
3155                                    ;       EFT - ERROR FORMAT TABLE ADDRESS
3156                                    ;
3157                                    ;THE EMT ENTRY IS THE ADDRESS OF THE TABLE OF ERROR MESSAGE STRINGS
3158                                    ;FOR THE PARTICULAR ERROR.  EACH ERROR MESSAGE TABLE LISTS THE ADDRESS
```

```
3159                    ;OF ONE OR MORE ERROR MESSAGE STRINGS WHICH ARE TO BE FORMATTED AND
3160                    ;TYPED BY THE ERROR TYPE SUBROUTINE.  IF THE EMT ENTRY IS ZERO, THERE IS
3161                    ;NO MESSAGE TO BE TYPED FOR THE ERROR.
3162                    ;
3163                    ;SIMILARLY, THE EHT, EDT, AND EFT ENTRIES ARE ADDRESSES OF TABLES
3164                    ;OF HEADER, DATA AND FORMAT INFORMATION FOR A GIVEN ERROR.  EACH ENTRY
3165                    ;IN THE ERROR HEADER TABLE MAY OR MAY NOT HAVE AN ASSOCIATED LINE OF
3166                    ;DATA, HOWEVER, EACH DATA LINE MUST HAVE AN ASSOCIATED FORMAT AND
3167                    ;HEADER. THAT IS, A HEADER LINE MAY BE PRINTED WITHOUT ANY DATA,
3168                    ;BUT A DATA LINE IS NOT PRINTED WITHOUT A HEADER, AND EACH DATA LINE
3169                    ;MUST ALSO HAVE A FORMAT.
3170                    ;
3171                    ;IN SUMMARY,
3172                    ;
3173                    ;       EACH NONZERO ENTRY IS THE ADDRESS OF A TABLE,
3174                    ;       EACH TABLE IS A LIST OF ADDRESSES WHICH DEFINES THE LOCATIONS
3175                    ;OF MESSAGE STRINGS, HEADERS, DATA OR FORMAT.
3176                    ;
3177
```

```
                                        .SBTTL  START OF PROGRAM
                                        ;
                                        START:
                                        ;CLEAR AND SETUP FOR TEST EXECUTION
                                                NOP
                                                RESET                   ;INITIALIZE THE SYSTEM
                                                MOV     PR6,-(SP)       ;;PUT NEW PS ON STACK
                                                MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
                                                RTI                     ;;POP NEW PC AND PS
                                        64$:
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
                                                MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
                                                CLR     (R6)+           ;;CLEAR MEMORY LOCATION
                                                CMP     #SWR,R6 ;;DONE?
                                                BNE     .-6             ;;LOOP BACK IF NO
                                                MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
                                                MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
                                                MOV     #340,@#IOTVEC+2 ;;LEVEL 7
                                                MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
                                                MOV     #340,@#EMTVEC+2 ;;LEVEL 7
                                                MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
                                                MOV     #340,@#TRAPVEC+2;LEVEL 7
                                                MOV     #$PWRON,@#PWRVEC ;;POWER FAILURE VECTOR
                                                MOV     #340,@#PWRVEC+2 ;;LEVEL 7
                                                MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
                                                CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
                                                CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
                                                MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
                                                MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
                                                MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
                                                MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
                                                MOV     #65$,@#ERRVEC   ;;SET UP ERROR VECTOR
                                                MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
                                                MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
                                                CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
                                                BNE     67$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
                                                BR      66$             ;;BRANCH IF NO TIMEOUT
                                        65$:    MOV     #66$,(SP)       ;;SET UP FOR TRAP RETURN
                                                RTI
                                        66$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
                                                MOV     #DISPREG,DISPLAY
                                        67$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

                                                CLR     $PASS           ;;CLEAR PASS COUNT
                                                BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
                                                BEQ     68$             ;;YES,USE NON-APT SWITCH
                                                MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
                                        68$:
```

```
                                        .SBTTL  TYPE PROGRAM NAME
                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
005610  005227  177777                          INC     #-1             ;;FIRST TIME?
005614  001047                                  BNE     69$             ;;BRANCH IF NO
005616  022737  036436  000042                  CMP     #$ENDAD,@#42    ;;ACT-11?
005624  001443                                  BEQ     69$             ;;BRANCH IF YES
005626  104401  005674                          TYPE    ,70$            ;;TYPE ASCIZ STRING
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
005632  005737  000042                          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
005636  001012                                  BNE     71$             ;;BRANCH IF YES
005640  123727  001242  000001                  CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
005646  001406                                  BEQ     71$             ;;BRANCH IF YES
005650  023727  001154  000176                  CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
005656  001005                                  BNE     72$             ;;BRANCH IF NO
005660  104407                                  GTSWR                   ;;GET SOFT-SWR SETTINGS
005662  000403                                  BR      72$
005664  112737  000001  001150          71$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
005672                                  72$:
005672  000420                                  BR      69$             ;;GET OVER THE ASCIZ
                                        ;;70$:  .ASCIZ  <CRLF>@RM03 FUNCTIONAL TEST, PART 3@<CRLF>
005734                                  69$:


                                        ;FIND OUT IF PROGRAM IS RUNNING IN STANDALONE MODE
005734  005737  000042                          TST     42              ;IS LOC 42 ZERO ??
005740  001003                                  BNE     10$             ;NO - NOT IN STANDALONE
005742  105737  001242                          TSTB    $ENV            ;IS APT ENVIRONMENT ZERO ??
005746  001411                                  BEQ     STANDALONE      ;YES - PROGRAM IN STANDALONE
005750                                  10$:


                                        ;PROGRAM NOT RUNNING IN STANDALONE - SEE IF SIZING IS ALLOWED
005750  132737  000200  001243                  BITB    #BIT7,$ENVM     ;SIZING ALLOWED ??
005756  001003                                  BNE     20$             ;NO
005760  012737  000377  001300                  MOV     #377,$DEVM      ;YES - SET DEVICE MAP FOR ALL DEVICES
005766                                  20$:


                                        ;GO TO COMMON START CODE
005766  000137  006610                          JMP     CMNSTART
```

```
3272  005772                          STANDALONE:
3273  005772  004737  064622                JSR     PC,$TKINT       ;INITIALIZE CONSOLE
3274
3275                                  ;SEE IF THIS IS THE FIRST START AFTER PROGRAM WAS LOADED
3276  005776  005327  000001                DEC     #1              ;FIRST START ??
3277  006002  100024                        BPL     10$             ;YES !!
3278
3279
3280                                  ;SEE IF THE USER WANTS TO KEEP SAME DEVICES FOR TESTING
3281  006004  104401  067154          5$:   TYPE    ,CNSL00         ;MAINTAIN PREVIOUS PARAMETERS??
3282  006010  104411                        RDCHR                   ;GET RESPONSE
3283  006012  012637  001176                MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3284  006016  104401  001176                TYPE    ,$TMP1
3285  006022  123727  001176  000131        CMPB    $TMP1,#'Y       ;YES RESPONSE??
3286  006030  001002                        BNE     6$              ;NO!!
3287  006032  000137  006742                JMP     READY           ;KEEP PREVIOUS PARAMETERS
3288  006036  123727  001176  000116  6$:   CMPB    $TMP1,#'N       ;NO RESPONSE??
3289  006044  001420                        BEQ     20$             ;GET NEW PARAMETERS
3290  006046  104401  067023                TYPE    ,QSTMRK         ;NOT YES OR NO, TYPE "?"
3291  006052  000754                        BR      5$              ;RETRY
3292  006054                          10$:
3293
3294                                  ;SEE IF OPERATOR WANTS HELP FILE
3295  006054  104401  067025                TYPE    ,HELPQST        ;WANT HELP ??
3296  006060  104411                        RDCHR                   ;GET RESPONSE
3297  006062  012637  001176                MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
3298  006066  104401  001176                TYPE    ,$TMP1
3299  006072  123727  001176  000131        CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
3300  006100  001002                        BNE     20$             ;NO - DONT TYPE HELP
3301  006102  104401  106520                TYPE    ,HELP           ;YES - TYPE HELP TEXT
3302  006106                          20$:
3303
3304                                  ;SEE IF USER WANTS TO CHANGE RM03 UNIBUS ADDRESS
3305  006106  104401  067061                TYPE    ,UBUSQST        ;WANT TO CHANGE ADDRESS ??
3306  006112  104411                        RDCHR                   ;GET RESPONSE
3307  006114  012637  001176                MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
3308  006120  104401  001176                TYPE    ,$TMP1
3309  006124  123727  001176  000131        CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
3310  006132  001137                        BNE     90$             ;NO !!
3311  006134                          30$:
3312
3313                                  ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, AND INTERRUPT VECTOR
3314  006134  104401  067213                TYPE    ,CNSL01         ;TYPE CURRENT BUS ADDRESS
3315  006140  013746  001276                MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
3316  006144  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3317  006146  104401  067014                TYPE    ,CLSPRN
3318  006152  104413                        RDOCT                   ;GET NEW BUS ADDRESS
3319  006154  012637  001176                MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
3320  006160  001412                        BEQ     50$             ;YES-SKIP TO NEXT ENTRY
3321  006162  022737  160000  001176        CMP     #160000,$TMP1   ;BASE ADDRESS IN I/O PAGE??
3322  006170  101403                        BLOS    40$             ;YES
3323  006172  104401  067240                TYPE    ,CNSL02         ;TYPE WARNING MESSAGE
3324  006176  000756                        BR      30$             ;RETRY
3325  006200  013737  001176  001276  40$:   MOV     $TMP1,$BASE     ;STORE NEW BUS ADDRESS
3326  006206  113737  001272  001176  50$:   MOVB    $VECT1,$TMP1    ;TYPE CURRENT VECTOR ADDRESS
3327  006214  105037  001177                CLRB    $TMP1+1
```

```
3328  006220  104401  067321                  TYPE    ,CNSL03
3329  006224  013746  001176                  MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
3330  006230  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3331  006232     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
3332  006233     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3333  006234  104401  067014                  TYPE    ,CLSPRN
3334  006240  104413                          RDOCT                   ;GET NEW VECTOR ADDRESS
3335  006242  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN?
3336  006246  001412                          BEQ     70$             ;YES-SKIP TO NEXT ENTRY
3337  006250  022737  001000  001176          CMP     #1000,$TMP1     ;VECTOR ADDRESS < 1000??
3338  006256  101003                          BHI     60$             ;YES!!
3339  006260  104401  067351                  TYPE    ,CNSL04         ;TYPE WARNING MESSAGE
3340  006264  000750                          BR      50$             ;RETRY
3341  006266  113737  001176  001272  60$:    MOVB    $TMP1,$VECT1    ;STORE NEW VECTOR ADDRESS
3342  006274  113737  001273  001176  70$:    MOVB    $VECT1+1,$TMP1  ;TYPE CURRENT PRIORITY
3343  006302  006237  001176                  ASR     $TMP1
3344  006306  006237  001176                  ASR     $TMP1
3345  006312  006237  001176                  ASR     $TMP1
3346  006316  006237  001176                  ASR     $TMP1
3347  006322  006237  001176                  ASR     $TMP1
3348  006326  105037  001177                  CLRB    $TMP1+1
3349  006332  104401  067425                  TYPE    ,CNSL05
3350  006336  013746  001176                  MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
3351  006342  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3352  006344     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
3353  006345     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3354  006346  104401  067014                  TYPE    ,CLSPRN
3355  006352  104413                          RDOCT                   ;GET NEW PRIORITY
3356  006354  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
3357  006360  001424                          BEQ     90$             ;YES-SKIP TO NEXT ENTRY
3358  006362  023727  001176  000007          CMP     $TMP1,#7        ;LEGAL PRIORITY??
3359  006370  002403                          BLT     80$             ;YES!!
3360  006372  104401  067461                  TYPE    ,CNSL06         ;TYPE WARNING MESSAGE
3361  006376  000736                          BR      70$             ;RETRY
3362  006400                          80$:                            ;STORE NEW PRIORITY
3363  006400  006337  001176                  ASL     $TMP1
3364  006404  006337  001176                  ASL     $TMP1
3365  006410  006337  001176                  ASL     $TMP1
3366  006414  006337  001176                  ASL     $TMP1
3367  006420  006337  001176                  ASL     $TMP1
3368  006424  113737  001176  001273          MOVB    $TMP1,$VECT1+1
3369  006432                          90$:
3370
3371                                  ;DIALOGUE TO INPUT DEVICE NUMBERS
3372  006432  005037  001300                  CLR     $DEVM           ;CLEAR DEVICE MAP
3373  006436  104401  067506                  TYPE    ,CNSL07         ;TYPE INPUT INSTRUCTIONS
3374  006442  104401  067017                  TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
3375  006446  104411                          RDCHR                   ;GET RESPONSE
3376  006450  012637  001176                  MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3377  006454  104401  001176                  TYPE    ,$TMP1
3378  006460  023727  001176  000101          CMP     $TMP1,#'A       ;TEST ALL DRIVES??
3379  006466  001017                          BNE     110$            ;NO
3380  006470  012737  000377  001300          MOV     #377,$DEVM      ;TEST ALL DEVICES
3381  006476  000444                          BR      140$            ;SKIP TO NEXT ENTRY
3382  006500  104401  067017          100$:   TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
3383  006504  104411                          RDCHR                   ;GET RESPONSE
```

```
3384  006506  012637  001176                    MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3385  006512  104401  001176                    TYPE    ,$TMP1
3386  006516  023727  001176  000015            CMP     $TMP1,#CR       ;CARRIAGE RETURN??
3387  006524  001431                            BEQ     140$
3388  006526  023727  001176  000060  110$:     CMP     $TMP1,#'0       ;NUMBER < 0??
3389  006534  002404                            BLT     120$            ;YES!!
3390  006536  023727  001176  000067            CMP     $TMP1,#'7       ;NUMBER > 7??
3391  006544  003403                            BLE     130$            ;NO!!
3392  006546  104401  067023            120$:   TYPE    ,QSTMRK         ;TYPE "?"
3393  006552  000752                            BR      100$            ;RETRY
3394  006554  013701  001176            130$:   MOV     $TMP1,R1        ;R1=DRIVE NUMBER
3395  006560  042701  177770                    BIC     #^C7,R1
3396  006564  116102  067746                    MOVB    ATNTBL(R1),R2           ;DECODE DEVICE NUMBER
3397  006570  042702  177400                    BIC     #^C377,R2               ;CLEAR UNUSED BITS
3398  006574  050237  001300                    BIS     R2,$DEVM                ;SET DEVICE # IN MAP
3399  006600  122737  000377  001300            CMPB    #377,$DEVM      ;DONE ??
3400  006606  101334                            BHI     100$            ;NO
3401  006610                           140$:
3402
```

```
 3403  006610                          CMNSTART:
 3404
 3405                                  ;ASSEMBLE TEST QUE FROM DEVICE MAP
 3406  006610  013700  001300                  MOV     $DEVM,R0        ;R0 = DEVICE MAP
 3407  006614  012701  001450                  MOV     #TSTQUE+2,R1    ;R1 = ADDRESS OF FIRST ENTRY IN QUE
 3408  006620  010137  001446                  MOV     R1,TSTQUE       ;INITIALIZE ENTRY POINTER
 3409  006624  012702  000001                  MOV     #1,R2           ;R2 = DEVICE POINTER
 3410  006630  005003                          CLR     R3              ;R3 = DEVICE NUMBER
 3411  006632  030200                  10$:    BIT     R2,R0           ;IS THIS DEVICE IN MAP ??
 3412  006634  001406                          BEQ     20$             ;NO !!
 3413  006636  010311                          MOV     R3,(R1)         ;YES - ENTER DEVICE NUMBER IN QUE
 3414  006640  116361  067746  000001          MOVB    ATNTBL(R3),1(R1);ENTER ATTENTION BIT IN QUE
 3415  006646  062701  000002                  ADD     #2,R1           ;ADVANCE ENTRY POINTER
 3416  006652  006302                  20$:    ASL     R2              ;ADVANCE DEVICE POINTER
 3417  006654  105702                          TSTB    R2              ;DONE ALL DEVICES ??
 3418  006656  001402                          BEQ     25$             ;YES
 3419  006660  005203                          INC     R3              ;ADVANCE DEVICE NUMBER
 3420  006662  000763                          BR      10$             ;ENTER NEXT DEVICE
 3421  006664  005011                  25$:    CLR     (R1)            ;TERMINATE TEST QUE
 3422
 3423                                  ;SIZE FOR CLOCK
 3424  006666  004737  043464                  JSR     PC,SIZCLK       ;SEE IF CLOCK PRESENT
 3425  006672  000403                          BR      40$             ;YES - CLOCK IS PRESENT
 3426  006674  104000                  30$:    ERROR                   ;NO CLOCK
 3427  006676  000000                          HALT
 3428  006700  000775                          BR      30$
 3429  006702                          40$:
 3430                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
 3431  006702  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
 3432  006706  001012                          BNE     64$             ;;BRANCH IF YES
 3433  006710  123727  001242  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
 3434  006716  001406                          BEQ     64$             ;;BRANCH IF YES
 3435  006720  023727  001154  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
 3436  006726  001005                          BNE     65$             ;;BRANCH IF NO
 3437  006730  104407                          GTSWR                   ;;GET SOFT-SWR SETTINGS
 3438  006732  000403                          BR      65$
 3439  006734  112737  000001  001150  64$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
 3440  006742                          65$:
 3441  006742  000240                  READY:  NOP                     ;READY TO START TEST
 3442  006744  004737  064622                  JSR     PC,$TKINT       ;INITIALIZE TTY
 3443  006750  013746  000300                  MOV     PR6,-(SP)       ;;PUT NEW PS ON STACK
 3444  006754  012746  006762                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
 3445  006760  000002                          RTI                     ;;POP NEW PC AND PS
 3446  006762                          64$:
 3447  006762  117737  172460  001234          MOVB    @TSTQUE,$UNIT   ;LOAD UNIT NUMBER
 3448  006770  005037  001472                  CLR     MEDENB          ;CLEAR MEDIA ENABLE
```

```
                                ;;*****************************************************************
                                ;*TEST 1         CONTROLLER ACCESS TEST
                                ;;*****************************************************************
006774                          TST1:
006774  000240                          NOP                     ;START OF TEST
006776  012737  007012  001122          MOV     #1$,$LPADR
007004  012737  007012  001124          MOV     #1$,$LPERR
007012                          1$:
007012  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
007016  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
007022  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
007026  012737  000001  001226          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
007034  005001                          CLR     R1
007036  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
007042  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
007046  012737  007140  000004          MOV     #3$,ERRVEC
007054  012737  000300  000006          MOV     #PR6,ERRVEC+2

007062  110160  000001                  MOVB    R1,RMCS1+1(R0)  ;MOVE HI BYTE TO RMCS1
007066  010160  000002                  MOV     R1,RMWC(R0)     ;MOVE WORD COUNT REGISTER
007072  016002  000002                  MOV     RMWC(R0),R2
007076  010160  000004                  MOV     R1,RMBA(R0)     ;MOVE BUS ADDRESS REGISTER
007102  016002  000004                  MOV     RMBA(R0),R2
007106  010160  000010                  MOV     R1,RMCS2(R0)    ;MOVE CONTROL STATUS REGISTER
007112  016002  000010                  MOV     RMCS2(R0),R2
007116  010160  000022                  MOV     R1,RMDB(R0)     ;MOVE DATA BUFFER
007122  016002  000022                  MOV     RMDB(R0),R2
007126  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
007132  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
007136  000415                          BR      7$              ;NO BUS TIMEOUT OCCURRED

007140  022626                  3$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
007142  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
007146  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
007152  104110                          ERROR   110             ;CANNOT ACCESS MASSBUS CONTROLLER
007154  005737  000042                  TST     42              ;STAND ALONE MODE??
007160  001002                          BNE     5$              ;NO!!
007162  000137  005322                  JMP     START           ;YES-GO GET $BASE
007166  000137  036246          5$:     JMP     $EOP            ;GO TO END OF PASS HANDLER
007172                          7$:

                                ;;*****************************************************************
                                ;*TEST 2         DEVICE AVAILABLE TEST

                                ;;*****************************************************************
007172                          TST2:
007172  000004                          SCOPE                   ;SCOPE CALL
007174  000240                          NOP                     ;START OF TEST
007176  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
007202  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
007206  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
007212  012737  000002  001226          MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX

007220  004737  052140                  JSR     PC,CNTCLR
007224  000404                          BR      2$      ;GO TO 2$ IF NO ERROR
007226  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
3505  007230  104000                        ERROR                           ;ERROR NUMBER DEFINED BY SUB
3506  007232  000137  007352                JMP     7$      ;GO TO 7$ IF ERROR WAS FOUND
3507  007236                        2$:
3508  007236  013746  000004                MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
3509  007242  013746  000006                MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
3510  007246  012737  007336  000004        MOV     #5$,ERRVEC
3511  007254  013737  000300  000006        MOV     PR6,ERRVEC+2
3512
3513  007262  016037  000000  001176        MOV     RMCS1(R0),$TMP1 ;GET DVA STATUS
3514  007270  016037  000010  001174        MOV     RMCS2(R0),$TMP0 ;GET NED STATUS
3515  007276  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
3516  007302  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
3517  007306  032737  010000  001174        BIT     #NED,$TMP0      ;NONEXISTENT DEVICE??
3518  007314  001402                        BEQ     3$              ;NO!!
3519  007316  104111                        ERROR   111             ;NONEXISTENT DEVICE
3520  007320  000414                        BR      7$
3521  007322  032737  004000  001176  3$:   BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
3522  007330  001012                        BNE     9$              ;YES!!
3523  007332  104112                        ERROR   112             ;DEVICE NOT AVAILABLE
3524  007334  000406                        BR      7$
3525
3526  007336  022626                5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
3527  007340  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
3528  007344  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
3529  007350  104113                        ERROR   113             ;BUS TIMEOUT (04 TRAP)
3530  007352  000137  036212        7$:     JMP     $EOSP
3531
3532  007356                        9$:
3533
3534                                ;;**********************************************************************
3535                                ;*TEST 3         DRIVE TYPE TEST
3536
3537                                ;;**********************************************************************
3538  007356                        TST3:
3539  007356  000004                        SCOPE                           ;SCOPE CALL
3540  007360  000240                        NOP                             ;START OF TEST
3541  007362  012706  001100                MOV     #STACK,SP               ;INITIALIZE STACK POINTER
3542  007366  013700  001276                MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
3543  007372  013701  001446                MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
3544  007376  012737  000003  001226        MOV     #3,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
3545
3546  007404  004737  052140                JSR     PC,CNTCLR
3547  007410  000404                        BR      2$      ;GO TO 2$ IF NO ERROR
3548  007412  000240                        NOP                             ;RETURN HERE IF ERROR
3549  007414  104000                        ERROR                           ;ERROR NUMBER DEFINED BY SUB
3550  007416  000137  007514                JMP     4$      ;GO TO 4$ IF ERROR WAS FOUND
3551  007422                        2$:
3552  007422  112737  000026  001504        MOVB    #RMDT,GETINX    ;SETUP GET INDEX TABLE
3553  007430  112737  000200  001505        MOVB    #200,GETINX+1
3554  007436  012737  007520  001354        MOV     #5$,RMDTI               ;RMDT INPUT BUFFER = 5$
3555  007444  004737  042776                JSR     PC,GET                  ;GO GET DRIVE TYPE
3556  007450  000402                        BR      3$      ;GO TO 3$ IF NO ERROR
3557  007452  000240                        NOP                             ;RETURN HERE IF ERROR
3558  007454  104000                        ERROR                           ;ERROR NUMBER SPECIFIED BY GET
3559  007456  022737  020024  001354  3$:   CMP     #SNGPRT,RMDTI   ;SINGLE PORT RM03??
3560  007464  001415                        BEQ     5$              ;YES!!
```

```
3561  007466  022737  024024  001354          CMP     #DULPRT,RMDTI   ;DUAL PORT RM03??
3562  007474  001411                          BEQ     5$              ;YES!!
3563  007476  012737  020024  001176          MOV     #SNGPRT,$TMP1
3564  007504  012737  024024  001200          MOV     #DULPRT,$TMP2
3565  007512  104114                          ERROR   114             ;NOT AN RM03
3566  007514  000137  036212          4$:     JMP     $EOSP           ;GO TO SUBPASS HANDLER.
3567
3568  007520                          5$:
3569                                  ;;*****************************************************************
3570                                  ;*TEST 4         WRITE, READ ZEROS
3571                                  ;;*****************************************************************
3572  007520                          TST4:
3573  007520  000004                          SCOPE                   ;SCOPE CALL
3574  007522  000240                          NOP                     ;START OF TEST
3575  007524  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3576  007530  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3577  007534  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3578  007540  012737  000004  001226          MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3579
3580                                  ;;*****************************************************************
3581                                  ;LOOP #1        FORMAT,WRITE,READ
3582
3583  007546                          10$:
3584
3585                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
3586  007546  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
3587  007552  054130                          .WORD   054130  ;TASK DESCRIPTOR
3588  007554  000404                          BR      20$             ;GO TO 20$ IF NO ERROR
3589  007556  000240                          NOP                     ;RETURN HERE IF ERROR
3590  007560  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3591  007562  000137  010542                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
3592  007566                          20$:
3593
3594                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
3595  007566  012737  000000  001432          MOV     #0,RMDC0        ;CYLINDER = 0
3596  007574  012737  000000  001404          MOV     #0,RMDA0        ;TRACK = 0, SECTOR = 0
3597  007602  012737  106520  001402          MOV     #BUFONE,RMBA0   ;BUS ADDRESS
3598  007610  012737  177376  001400          MOV     #<-C<2+256.>+1>,RMWC0           ;WORD COUNT = 1 SECTOR
3599  007616  012737  010000  001430          MOV     #FMT16,RM0F0    ;16 BIT FORMAT
3600  007624  012737  000063  001376          MOV     #WH!GO,RMCS10   ;WRITE HEADER AND DATA COMMAND
3601
3602                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
3603
3604  007632  004737  040174                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
3605  007636  000405                          BR      25$             ;GO TO 25$ IF NO ERROR
3606  007640  104401  066704                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
3607  007644  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
3608  007646  000137  010542                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
3609  007652                          25$:
3610  007652  012737  070060  001174          MOV     #ZEROS,$TMP0    ;STARTING ADDRESS OF PATTERN
3611  007660  012737  000001  001176          MOV     #1,$TMP1        ;RANGE OF PATTERN
3612  007666  004737  042100                  JSR     PC,GENBUF       ;GO GENERATE BUFFER FOR FORMAT
3613
3614                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
3615  007672  012702  001533                  MOV     #PUTINX,R2      ;R2 = BYTE ENTRY POSITION
3616  007676  112722  000034                  MOVB    #RMDC,(R2)+
```

```
3617  007702  112722  000006                  MOVB    #RMDA,(R2)+
3618  007706  112722  000004                  MOVB    #RMBA,(R2)+
3619  007712  112722  000002                  MOVB    #RMWC,(R2)+
3620  007716  112722  000032                  MOVB    #RMOF,(R2)+
3621  007722  112722  000000                  MOVB    #RMCS1,(R2)+
3622  007726  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
3623  007732                          30$:
3624
3625                                  ;FORMAT THE DRIVE
3626  007732  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
3627  007736  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
3628  007740  000240                          NOP                     ;RETURN HERE IF ERROR
3629  007742  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
3630  007744  000137  010542                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3631  007750                          40$:
3632
3633                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
3634  007750  004737  042712                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
3635
3636                                  ;WAIT FOR THE FORMAT TO COMPLETE
3637  007754  004737  043606                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
3638
3639                                  ;GO READ STATUS FOR FORMAT OPERATION
3640  007760  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
3641  007764  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
3642  007766  000240                          NOP                     ;RETURN HERE IF ERROR
3643  007770  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
3644  007772  000137  010542                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3645  007776                          50$:
3646
3647                                  ;VERIFY NO ERRORS DURING FORMAT
3648  007776  004737  056276                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
3649  010002  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
3650  010004  000240                          NOP                     ;RETURN HERE IF ERROR
3651  010006  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3652  010010  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
3653  010012  000137  010542                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3654  010016                          60$:
3655
3656                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
3657  010016  012737  010034  001122          MOV     #70$,$LPADR
3658  010024  012737  010034  001124          MOV     #70$,$LPERR
3659  010032  000410                          BR      80$                             ;SKIP TO WRITE OPERATION
3660
3661                                  ;;*****************************************************************
3662                                  ;LOOP #2        WRITE,READ
3663
3664  010034                          70$:
3665
3666
3667                                  ;PREPARE DEVICE FOR WRITE OPERATION
3668  010034  004737  037260                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
3669  010040  054130                          .WORD   054130  ;TASK DESCRIPTOR
3670  010042  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
3671  010044  000240                          NOP                     ;RETURN HERE IF ERROR
3672  010046  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
3673  010050  000137  010542                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3674  010054                                80$:
3675
3676
3677  010054                                120$:
3678
3679                                        ;WRITE DATA TO THE DRIVE
3680  010054  012737  106524  001402              MOV     #BUFONE+4,RMBA0         ;MOVE MEMORY ADDRESS
3681  010062  012737  177400  001400              MOV     #<^C256.+1>,RMWC0       ;CHANGE WORD COUNT
3682  010070  012737  000061  001376              MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
3683  010076  012702  001533                      MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
3684  010102  112722  000006                      MOVB    #RMDA,(R2)+
3685  010106  112722  000034                      MOVB    #RMDC,(R2)+
3686  010112  112722  000032                      MOVB    #RMOF,(R2)+
3687  010116  112722  000004                      MOVB    #RMBA,(R2)+
3688  010122  112722  000002                      MOVB    #RMWC,(R2)+
3689  010126  112722  000000                      MOVB    #RMCS1,(R2)+
3690  010132  112722  000200                      MOVB    #200,(R2)+              ;TERMINATE TABLE
3691
3692  010136  004737  043246                      JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
3693  010142  000404                              BR      130$                    ;GO TO 130$ IF NO ERROR
3694  010144  000240                              NOP                     ;RETURN HERE IF ERROR
3695  010146  104000                              ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
3696  010150  000137  010542                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3697  010154                                130$:
3698
3699                                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
3700  010154  004737  042712                      JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
3701
3702                                        ;WAIT FOR WRITE COMMAND TO COMPLETE
3703  010160  004737  043606                      JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
3704
3705                                        ;GO READ STATUS FOR WRITE COMMAND
3706  010164  004737  042776                      JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
3707  010170  000404                              BR      140$                    ;GO TO 140$ IF NO ERROR
3708  010172  000240                              NOP                     ;RETURN HERE IF ERROR
3709  010174  104000                              ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
3710  010176  000137  010542                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3711  010202                                140$:
3712
3713                                        ;CHECK FOR ERRORS DURING WRITE OPERATION
3714  010202  004737  043772                      JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
3715  010206  000405                              BR      150$                    ;GO TO 150$ IF NO ERROR
3716  010210  000240                              NOP                     ;RETURN HERE IF ERROR
3717  010212  104000                              ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
3718  010214  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3719  010216  000137  010542                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3720  010222                                150$:
3721  010222  004737  056276                      JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
3722  010226  000405                              BR      160$                    ;GO TO 160$ IF NO ERROR
3723  010230  000240                              NOP                     ;RETURN HERE IF ERROR
3724  010232  104000                              ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3725  010234  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3726  010236  000137  010542                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3727  010242                                160$:
3728
```

```
3729  010242                          170$:
3730  010242  004737  044624                  JSR     PC,$ECERR       ;GO CHECK FOR SECONDARY ERRORS
3731  010246  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
3732  010250  000240                          NOP                     ;RETURN HERE IF ERROR
3733  010252  104000                          ERROR                   ;ERROR # DEFINED BY $ECERR SUBROUTINE
3734  010254  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3735  010256  000137  010542                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND)
3736  010262                          180$:
3737
3738                                  ;CHANGE LOOP ADDRESSES
3739  010262  012737  010300  001122          MOV     #190$,$LPADR
3740  010270  012737  010300  001124          MOV     #190$,$LPERR
3741  010276  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
3742
3743                                  ;;*****************************************************************
3744                                  ;LOOP #3        READ
3745
3746  010300                          190$:
3747
3748                                  ;PREPARE DEVICE FOR READ OPERATION
3749  010300  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
3750  010304  054130                          .WORD   054130  ;TASK DESCRIPTOR
3751  010306  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
3752  010310  000240                          NOP                     ;RETURN HERE IF ERROR
3753  010312  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3754  010314  000137  010542                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3755  010320                          200$:
3756
3757  010320                          240$:
3758
3759                                  ;READ DATA FROM DEVICE
3760  010320  012737  107530  001402          MOV     #BUFTWO+4,RMBAO         ;CHANGE MEMORY ADDRESS
3761  010326  012737  000071  001376          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
3762  010334  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
3763  010340  112722  000006                  MOVB    #RMDA,(R2)+
3764  010344  112722  000032                  MOVB    #RMOF,(R2)+
3765  010350  112722  000034                  MOVB    #RMDC,(R2)+
3766  010354  112722  000004                  MOVB    #RMBA,(R2)+
3767  010360  112722  000002                  MOVB    #RMWC,(R2)+
3768  010364  112722  000000                  MOVB    #RMCS1,(R2)+
3769  010370  112712  000200                  MOVB    #200,(R2)
3770
3771  010374  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
3772  010400  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
3773  010402  000240                          NOP                     ;RETURN HERE IF ERROR
3774  010404  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
3775  010406  000137  010542                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3776  010412                          250$:
3777
3778                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
3779  010412  004737  042712                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
3780
3781                                  ;WAIT FOR READ OPERATION TO COMPLETE
3782  010416  004737  043606                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
3783
3784                                  ;GO READ STATUS FOR READ OPERATION
```

```
3785  010422  004737  042776          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
3786  010426  000404                  BR      260$                    ;GO TO 260$ IF NO ERROR
3787  010430  000240                  NOP                     ;RETURN HERE IF ERROR
3788  010432  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
3789  010434  000137  010542          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3790  010440                  260$:
3791
3792                          ;CHECK FOR ERRORS DURING READ OPERATION
3793  010440  004737  043772          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
3794  010444  000405                  BR      270$                    ;GO TO 270$ IF NO ERROR
3795  010446  000240                  NOP                     ;RETURN HERE IF ERROR
3796  010450  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
3797  010452  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3798  010454  000137  010542          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3799  010460                  270$:
3800  010460  004737  056276          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
3801  010464  000405                  BR      280$                    ;GO TO 280$ IF NO ERROR
3802  010466  000240                  NOP                     ;RETURN HERE IF ERROR
3803  010470  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3804  010472  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3805  010474  000137  010542          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3806  010500                  280$:
3807
3808  010500                  290$:
3809  010500  004737  044624          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
3810  010504  000405                  BR      300$                    ;GO TO 300$ IF NO ERROR
3811  010506  000240                  NOP                     ;RETURN HERE IF ERROR
3812  010510  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
3813  010512  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3814  010514  000137  010542          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3815  010520                  300$:
3816  010520  004737  042344          JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
3817  010524  106524                  .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
3818  010526  107530                  .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
3819  010530  000404                  BR      310$                    ;GO TO 310$ IF NO ERROR
3820  010532  000240                  NOP                     ;RETURN HERE IF ERROR
3821  010534  104000                  ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
3822  010536  000137  010542          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3823  010542                  310$:
3824
3825  010542                  350$:
3826                          ;;*****************************************************************
3827                          ;*TEST 5         WRITE, WRITE CHECK ZEROS
3828                          ;;*****************************************************************
3829  010542                  TST5:
3830  010542  000004                  SCOPE                   ;SCOPE CALL
3831  010544  000240                  NOP                     ;START OF TEST
3832  010546  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3833  010552  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3834  010556  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3835  010562  012737  000005  001226  MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3836
3837                          ;;*****************************************************************
3838                          ;LOOP #1        FORMAT,WRITE,WRITE CHECK DATA
3839
3840  010570                  10$:
```

```
3841
3842                            ;PREPARE THE DEVICE FOR FORMAT OPERATION
3843  010570  004737  037260              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
3844  010574  054130                      .WORD   054130  ;TASK DESCRIPTOR
3845  010576  000404                      BR      20$                     ;GO TO 20$ IF NO ERROR
3846  010600  000240                      NOP                     ;RETURN HERE IF ERROR
3847  010602  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3848  010604  000137  011534              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3849  010610                          20$:
3850
3851                            ;LOAD PARAMETERS AND GENERATE DATA BUFFER
3852  010610  012737  000000  001432      MOV     #0,RMDC0                ;CYLINDER = 0
3853  010616  012737  000000  001404      MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
3854  010624  012737  106520  001402      MOV     #BUFONE,RMBA0           ;BUS ADDRESS
3855  010632  012737  177376  001400      MOV     #^C<2+256.>+1>,RMWC0            ;WORD COUNT = 1 SECTOR
3856  010640  012737  010000  001430      MOV     #FMT16,RMOF0            ;16 BIT FORMAT
3857  010646  012737  000063  001376      MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
3858
3859                            ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
3860
3861  010654  004737  040174              JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
3862  010660  000405                      BR      25$                     ;GO TO 25$ IF NO ERROR
3863  010662  104401  066704              TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
3864  010666  104000                      ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
3865  010670  000137  011534              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3866  010674                          25$:
3867  010674  012737  070060  001174      MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
3868  010702  012737  000001  001176      MOV     #1,$TMP1                ;RANGE OF PATTERN
3869  010710  004737  042100              JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
3870
3871                            ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
3872  010714  012702  001533              MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
3873  010720  112722  000034              MOVB    #RMDC,(R2)+
3874  010724  112722  000006              MOVB    #RMDA,(R2)+
3875  010730  112722  000004              MOVB    #RMBA,(R2)+
3876  010734  112722  000002              MOVB    #RMWC,(R2)+
3877  010740  112722  000032              MOVB    #RMOF,(R2)+
3878  010744  112722  000000              MOVB    #RMCS1,(R2)+
3879  010750  112712  000200              MOVB    #200,(R2)               ;TERMINATE TABLE
3880  010754                          30$:
3881
3882                            ;FORMAT THE DRIVE
3883  010754  004737  043246              JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
3884  010760  000404                      BR      40$                     ;GO TO 40$ IF NO ERROR
3885  010762  000240                      NOP                     ;RETURN HERE IF ERROR
3886  010764  104000                      ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
3887  010766  000137  011534              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3888  010772                          40$:
3889
3890                            ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
3891  010772  004737  042712              JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
3892
3893                            ;WAIT FOR THE FORMAT TO COMPLETE
3894  010776  004737  043606              JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
3895
3896                            ;GO READ STATUS FOR FORMAT OPERATION
```

```
3897  011002  004737  042776          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
3898  011006  000404                  BR      50$                     ;GO TO 50$ IF NO ERROR
3899  011010  000240                  NOP                     ;RETURN HERE IF ERROR
3900  011012  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
3901  011014  000137  011534          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3902  011020                  50$:
3903
3904                          ;VERIFY NO ERRORS DURING FORMAT
3905  011020  004737  056276          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
3906  011024  000405                  BR      60$                     ;GO TO 60$ IF NO ERROR
3907  011026  000240                  NOP                     ;RETURN HERE IF ERROR
3908  011030  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3909  011032  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3910  011034  000137  011534          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3911  011040                  60$:
3912
3913                          ;MOVE LOOP ADDRESSES TO NEXT OPERATION
3914  011040  012737  011056  001122          MOV     #70$,$LPADR
3915  011046  012737  011056  001124          MOV     #70$,$LPERR
3916  011054  000410                  BR      80$                     ;SKIP TO WRITE OPERATION
3917
3918                          ;;*****************************************************************
3919                          ;LOOP #2        WRITE,WRITE CHECK DATA
3920
3921  011056                  70$:
3922
3923
3924                          ;PREPARE DEVICE FOR WRITE OPERATION
3925  011056  004737  037260          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
3926  011062  054130                  .WORD   054130  ;TASK DESCRIPTOR
3927  011064  000404                  BR      80$                     ;GO TO 80$ IF NO ERROR
3928  011066  000240                  NOP                     ;RETURN HERE IF ERROR
3929  011070  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3930  011072  000137  011534          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3931  011076                  80$:
3932
3933
3934  011076                  120$:
3935
3936                          ;WRITE DATA TO THE DRIVE
3937  011076  012737  106524  001402          MOV     #BUFONE+4,RMBA0         ;CHANGE MEMORY ADDRESS
3938  011104  012737  177400  001400          MOV     #<^C256.+1>,RMWC0       ;CHANGE WORD COUNT
3939  011112  012737  000061  001376          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
3940  011120  012702  001533          MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
3941  011124  112722  000006          MOVB    #RMDA,(R2)+
3942  011130  112722  000034          MOVB    #RMDC,(R2)+
3943  011134  112722  000032          MOVB    #RMOF,(R2)+
3944  011140  112722  000004          MOVB    #RMBA,(R2)+
3945  011144  112722  000002          MOVB    #RMWC,(R2)+
3946  011150  112722  000000          MOVB    #RMCS1,(R2)+
3947  011154  112722  000200          MOVB    #200,(R2)+              ;TERMINATE TABLE
3948
3949  011160  004737  043246          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
3950  011164  000404                  BR      130$                    ;GO TO 130$ IF NO ERROR
3951  011166  000240                  NOP                     ;RETURN HERE IF ERROR
3952  011170  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
```

```
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
130$:

                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                ;WAIT FOR WRITE COMMAND TO COMPLETE
                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                ;GO READ STATUS FOR WRITE COMMAND
                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
                        BR      140$                    ;GO TO 140$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
140$:

                ;CHECK FOR ERRORS DURING WRITE OPERATION
                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                        BR      150$                    ;GO TO 150$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
150$:
                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
                        BR      160$                    ;GO TO 160$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
160$:

170$:
                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                        BR      180$                    ;GO TO 180$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
180$:

                ;CHANGE LOOP ADDRESSES
                        MOV     #190$,$LPADR
                        MOV     #190$,$LPERR
                        BR      200$                    ;SKIP TO NEXT OPERATION

                ;;*****************************************************************
                ;LOOP #3         WRITE CHECK DATA

190$:

                ;PREPARE DEVICE FOR WRITE CHECK DATA OPERATION
                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                        .WORD   054130  ;TASK DESCRIPTOR
                        BR      200$                    ;GO TO 200$ IF NO ERROR
```

```
                        NOP                             ;RETURN HERE IF ERROR
                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
011334  104000
011336  000137  011534  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
011342                  200$:

011342                  240$:

                        ;WRITE CHECK DATA DATA FROM DEVICE
011342  012737  000051  001376  MOV     #WCD!GO,RMCS10  ;WRITE CHECK DATA DATA COMMAND
011350  012702  001533          MOV     #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
011354  112722  000006          MOVB    #RMDA,(R2)+
011360  112722  000032          MOVB    #RMOF,(R2)+
011364  112722  000034          MOVB    #RMDC,(R2)+
011370  112722  000004          MOVB    #RMBA,(R2)+
011374  112722  000002          MOVB    #RMWC,(R2)+
011400  112722  000000          MOVB    #RMCS1,(R2)+
011404  112712  000200          MOVB    #200,(R2)

011410  004737  043246          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
011414  000404                  BR      250$            ;GO TO 250$ IF NO ERROR
011416  000240                  NOP                     ;RETURN HERE IF ERROR
011420  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
011422  000137  011534          JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
011426                  250$:

                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
011426  004737  042712          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                        ;WAIT FOR WRITE CHECK DATA OPERATION TO COMPLETE
011432  004737  043606          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                        ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
011436  004737  042776          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
011442  000404                  BR      260$            ;GO TO 260$ IF NO ERROR
011444  000240                  NOP                     ;RETURN HERE IF ERROR
011446  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
011450  000137  011534          JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
011454                  260$:

                        ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
011454  004737  043772          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
011460  000405                  BR      270$            ;GO TO 270$ IF NO ERROR
011462  000240                  NOP                     ;RETURN HERE IF ERROR
011464  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
011466  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
011470  000137  011534          JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
011474                  270$:
011474  004737  056276          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
011500  000405                  BR      280$            ;GO TO 280$ IF NO ERROR
011502  000240                  NOP                     ;RETURN HERE IF ERROR
011504  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
011506  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
011510  000137  011534          JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
011514                  280$:

011514                  290$:
```

```
4065  011514  004737  044624          JSR     PC,$ECERR       ;GO CHECK FOR SECONDARY ERRORS
4066  011520  000405                  BR      300$                    ;GO TO 300$ IF NO ERROR
4067  011522  000240                  NOP                     ;RETURN HERE IF ERROR
4068  011524  104000                  ERROR                   ;ERROR # DEFINED BY $ECERR SUBROUTINE
4069  011526  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4070  011530  000137  011534          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4071  011534                  300$:
4072
4073  011534                  350$:
4074                          ;;*****************************************************************
4075                          ;*TEST 6         WRITE, WRITE CHECK ZEROS W/ WCE ERROR
4076                          ;;*****************************************************************
4077  011534                  TST6:
4078  011534  000004                  SCOPE                   ;SCOPE CALL
4079  011536  000240                  NOP                     ;START OF TEST
4080  011540  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4081  011544  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4082  011550  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4083  011554  012737  000006  001226  MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4084
4085                          ;;*****************************************************************
4086                          ;LOOP #1        FORMAT,WRITE,WRITE CHECK DATA
4087
4088  011562                  10$:
4089
4090                          ;PREPARE THE DEVICE FOR FORMAT OPERATION
4091  011562  004737  037260          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4092  011566  054130                  .WORD   054130  ;TASK DESCRIPTOR
4093  011570  000404                  BR      20$                     ;GO TO 20$ IF NO ERROR
4094  011572  000240                  NOP                     ;RETURN HERE IF ERROR
4095  011574  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4096  011576  000137  012752          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4097  011602                  20$:
4098
4099                          ;LOAD PARAMETERS AND GENERATE DATA BUFFER
4100  011602  012737  000000  001432  MOV     #0,RMDC0                ;CYLINDER = 0
4101  011610  012737  000000  001404  MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
4102  011616  012737  106520  001402  MOV     #BUFONE,RMBA0           ;BUS ADDRESS
4103  011624  012737  177376  001400  MOV     #<^C<2+256.>+1>,RMWC0           ;WORD COUNT = 1 SECTOR
4104  011632  012737  010000  001430  MOV     #FMT16,RMOF0            ;16 BIT FORMAT
4105  011640  012737  000063  001376  MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
4106
4107                          ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
4108
4109  011646  004737  040174          JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
4110  011652  000405                  BR      25$                     ;GO TO 25$ IF NO ERROR
4111  011654  104401  066704          TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
4112  011660  104000                  ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
4113  011662  000137  012752          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4114  011666                  25$:
4115  011666  012737  070060  001174  MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
4116  011674  012737  000001  001176  MOV     #1,$TMP1                ;RANGE OF PATTERN
4117  011702  004737  042100          JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
4118
4119                          ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
4120  011706  012702  001533          MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
```

```
 4121  011712  112722  000034                MOVB    #RMDC,(R2)+
 4122  011716  112722  000006                MOVB    #RMDA,(R2)+
 4123  011722  112722  000004                MOVB    #RMBA,(R2)+
 4124  011726  112722  000002                MOVB    #RMWC,(R2)+
 4125  011732  112722  000032                MOVB    #RMOF,(R2)+
 4126  011736  112722  000000                MOVB    #RMCS1,(R2)+
 4127  011742  112712  000200                MOVB    #200,(R2)               ;TERMINATE TABLE
 4128  011746                            30$:
 4129
 4130                                    ;FORMAT THE DRIVE
 4131  011746  004737  043246                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
 4132  011752  000404                        BR      40$                     ;GO TO 40$ IF NO ERROR
 4133  011754  000240                        NOP                     ;RETURN HERE IF ERROR
 4134  011756  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
 4135  011760  000137  012752                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4136  011764                            40$:
 4137
 4138                                    ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
 4139  011764  004737  042712                JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
 4140
 4141                                    ;WAIT FOR THE FORMAT TO COMPLETE
 4142  011770  004737  043606                JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
 4143
 4144                                    ;GO READ STATUS FOR FORMAT OPERATION
 4145  011774  004737  042776                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 4146  012000  000404                        BR      50$                     ;GO TO 50$ IF NO ERROR
 4147  012002  000240                        NOP                     ;RETURN HERE IF ERROR
 4148  012004  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 4149  012006  000137  012752                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4150  012012                            50$:
 4151
 4152                                    ;VERIFY NO ERRORS DURING FORMAT
 4153  012012  004737  056276                JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
 4154  012016  000405                        BR      60$                     ;GO TO 60$ IF NO ERROR
 4155  012020  000240                        NOP                     ;RETURN HERE IF ERROR
 4156  012022  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 4157  012024  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 4158  012026  000137  012752                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4159  012032                            60$:
 4160
 4161                                    ;MOVE LOOP ADDRESSES TO NEXT OPERATION
 4162  012032  012737  012050  001122        MOV     #70$,$LPADR
 4163  012040  012737  012050  001124        MOV     #70$,$LPERR
 4164  012046  000410                        BR      80$                     ;SKIP TO WRITE OPERATION
 4165
 4166                                    ;;*****************************************************************
 4167                                    ;LOOP #2        WRITE,WRITE CHECK DATA
 4168
 4169  012050                            70$:
 4170
 4171
 4172                                    ;PREPARE DEVICE FOR WRITE OPERATION
 4173  012050  004737  037260                JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
 4174  012054  054130                        .WORD   054130  ;TASK DESCRIPTOR
 4175  012056  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
 4176  012060  000240                        NOP                     ;RETURN HERE IF ERROR
```

```
                                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                80$:

                                120$:

                                ;WRITE DATA TO THE DRIVE
                                        MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
                                        MOV     #BUFONE+4,RMBAO         ;CHANGE MEMORY ADDRESS
                                        MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
                                        MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
                                        MOVB    #RMDA,(R2)+
                                        MOVB    #RMDC,(R2)+
                                        MOVB    #RMOF,(R2)+
                                        MOVB    #RMBA,(R2)+
                                        MOVB    #RMWC,(R2)+
                                        MOVB    #RMCS1,(R2)+
                                        MOVB    #200,(R2)+              ;TERMINATE TABLE

                                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
                                        BR      130$                    ;GO TO 130$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                130$:

                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
                                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR WRITE COMMAND TO COMPLETE
                                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR WRITE COMMAND
                                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
                                        BR      140$                    ;GO TO 140$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                140$:

                                ;CHECK FOR ERRORS DURING WRITE OPERATION
                                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                                        BR      150$                    ;GO TO 150$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                150$:
                                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
                                        BR      160$                    ;GO TO 160$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                160$:
```

```
                                        170$:
012256  004737  044624                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
012262  000405                                BR      180$                    ;GO TO 180$ IF NO ERROR
012264  000240                                NOP                     ;RETURN HERE IF ERROR
012266  104000                                ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
012270  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
012272  000137  012752                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
012276                                  180$:

                                        ;CHANGE LOOP ADDRESSES
012276  012737  012320  001122                MOV     #190$,$LPADR
012304  012737  012320  001124                MOV     #190$,$LPERR
012312  012737  012324  001210                MOV     #195$,$ESCAPE   ;;ESCAPE TO 195$ ON ERROR

                                        ;;*****************************************************************
                                        ;LOOP #3        WRITE CHECK DATA

012320                                  190$:

012320  012703  000001                        MOV     #1,R3                   ;R3=WCE BIT POSITION
012324  050337  107522                195$:   BIS     R3,BUFTWO-2             ;CHANGE LAST WORD OF BUFFER

                                        ;PREPARE DEVICE FOR WRITE CHECK DATA OPERATION
012330  004737  037260                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
012334  054130                                .WORD   054130  ;TASK DESCRIPTOR
012336  000404                                BR      200$                    ;GO TO 200$ IF NO ERROR
012340  000240                                NOP                     ;RETURN HERE IF ERROR
012342  104000                                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
012344  000137  012752                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
012350                                  200$:

012350                                  240$:

                                        ;READ DATA FROM DEVICE
012350  012737  000051  001376                MOV     #WCD!GO,RMCS10          ;WRITE CHECK DATA DATA COMMAND
012356  012702  001533                        MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
012362  112722  000006                        MOVB    #RMDA,(R2)+
012366  112722  000032                        MOVB    #RMOF,(R2)+
012372  112722  000034                        MOVB    #RMDC,(R2)+
012376  112722  000004                        MOVB    #RMBA,(R2)+
012402  112722  000002                        MOVB    #RMWC,(R2)+
012406  112722  000000                        MOVB    #RMCS1,(R2)+
012412  112712  000200                        MOVB    #200,(R2)

012416  004737  043246                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
012422  000404                                BR      250$                    ;GO TO 250$ IF NO ERROR
012424  000240                                NOP                     ;RETURN HERE IF ERROR
012426  104000                                ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
012430  000137  012752                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
012434                                  250$:

                                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
012434  004737  042712                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                        ;WAIT FOR WRITE CHECK DATA OPERATION TO COMPLETE
```

```
4289  012440  004737  043606                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4290
4291                                ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
4292  012444  004737  042776                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4293  012450  000404                        BR      260$                    ;GO TO 260$ IF NO ERROR
4294  012452  000240                        NOP                     ;RETURN HERE IF ERROR
4295  012454  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4296  012456  000137  012752                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4297  012462                        260$:
4298
4299                                ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
4300  012462  004737  043772                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4301  012466  000405                        BR      270$                    ;GO TO 270$ IF NO ERROR
4302  012470  000240                        NOP                     ;RETURN HERE IF ERROR
4303  012472  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4304  012474  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4305  012476  000137  012752                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4306  012502                        270$:
4307  012502  032737  040000  001336        BIT     #WCE,RMCS2I             ;WAS "WCE" DETECTED??
4308  012510  001030                        BNE     285$                    ;YES!!
4309  012512  004737  056276                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4310  012516  000405                        BR      280$                    ;GO TO 280$ IF NO ERROR
4311  012520  000240                        NOP                     ;RETURN HERE IF ERROR
4312  012522  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4313  012524  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4314  012526  000137  012752                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4315  012532                        280$:
4316  012532  013737  001336  001140        MOV     RMCS2I,$GDDAT           ;EXPECTED STATUS
4317  012540  052737  040000  001140        BIS     #WCE,$GDDAT
4318  012546  013737  001336  001142        MOV     RMCS2I,$BDDAT           ;RECEIVED STATUS
4319  012554  010337  001174                MOV     R3,$TMP0                ;FAILING BIT POSITION
4320  012560  012737  107522  001176        MOV     #BUFTWO-2,$TMP1         ;FAILING ADDRESS
4321  012566  104337                        ERROR   337                     ;WCE NOT DETECTED
4322  012570  000470                        BR      350$
4323
4324  012572                        285$:
4325  012572  112737  000022  001504        MOVB    #RMDB,GETINX    ;SETUP GET INDEX TABLE
4326  012600  112737  000200  001505        MOVB    #200,GETINX+1
4327  012606  004737  042776                JSR     PC,GET          ;GO GET THE CONTENTS OF THE DATA BUFFER
4328  012612  000404                        BR      290$            ;GO TO 290$ IF NO ERROR
4329  012614  000240                        NOP                     ;RETURN HERE IF ERROR
4330  012616  104000                        ERROR                   ;ERROR DEFINED BY GET SUBROUTINE
4331  012620  000137  012752                JMP     350$            ;GO TO 350$ IF ERROR
4332
4333  012624                        290$:
4334  012624  013737  001350  001142        MOV     RMDBI,$BDDAT            ;RECEIVED DATA
4335  012632  013737  107522  001140        MOV     BUFTWO-2,$GDDAT         ;EXPECTED DATA
4336  012640  040337  001140                BIC     R3,$GDDAT
4337  012644  012737  107522  001134        MOV     #BUFTWO-2,$GDADR        ;EXPECTED ADDRESS
4338  012652  013737  001332  001136        MOV     RMBAI,$BDADR            ;RECEIVED ADDRESS
4339  012660  162737  000002  001136        SUB     #2,$BDADR               ;ADJUST BUS ADDRESS
4340  012666  023737  001134  001136        CMP     $GDADR,$BDADR           ;ADDRESSES OK??
4341  012674  001402                        BEQ     295$                    ;YES!!
4342  012676  104340                        ERROR   340                     ;WCE AT UNEXPECTED ADDRESS
4343  012700  000424                        BR      350$
4344  012702  023737  001140  001142  295$: CMP     $GDDAT,$BDDAT           ;DATA OK??
```

```
                        BEQ     296$                    ;YES!!
                        ERROR   341                     ;UNEXPECTED WCE DATA
                        BR      350$
                296$:
                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                        BR      300$                    ;GO TO 300$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                300$:
                        BIC     R3,BUFTWO-2             ;RESTORE DATA PATTERN
                        ASL     R3                      ;SHIFT TO NEXT BIT
                        BEQ     350$                    ;EXIT IF DONE
                        JMP     195$                    ;REPEAT TEST FOR NEXT DATA BIT
                350$:
                        MOV     #0,$ESCAPE      ;;ESCAPE TO 0 ON ERROR
                        MOV     #10$,$LPADR             ;CHANGE LOOP TO START OF TEST
                        MOV     #10$,$LPERR
                ;;******************************************************************
                ;*TEST 7         WRITE, READ ONES
                ;;******************************************************************
                TST7:
                        SCOPE                           ;SCOPE CALL
                        NOP                             ;START OF TEST
                        MOV     #STACK,SP               ;INITIALIZE STACK POINTER
                        MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
                        MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
                        MOV     #7,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX

                ;;******************************************************************
                ;LOOP #1         FORMAT,WRITE,READ

                10$:

                ;PREPARE THE DEVICE FOR FORMAT OPERATION
                        JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
                        .WORD   054130  ;TASK DESCRIPTOR
                        BR      20$                             ;GO TO 20$ IF NO ERROR
                        NOP                             ;RETURN HERE IF ERROR
                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                20$:

                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
                        MOV     #0,RMDC0                        ;CYLINDER = 0
                        MOV     #0,RMDA0                        ;TRACK = 0, SECTOR = 0
                        MOV     #BUFONE,RMBA0                   ;BUS ADDRESS
                        MOV     #<^C256.+1>,RMWC0               ;WORD COUNT = 1 SECTOR
                        MOV     #FMT16,RMOF0                    ;16 BIT FORMAT
                        MOV     #WH!GO,RMCS10                   ;WRITE HEADER AND DATA COMMAND

                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
```

```
 4401  013106  004737  040174            JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
 4402  013112  000405                    BR      25$                     ;GO TO 25$ IF NO ERROR
 4403  013114  104401  066704            TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
 4404  013120  104000                    ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
 4405  013122  000137  014016            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4406  013126                    25$:
 4407  013126  012737  070016  001174    MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
 4408  013134  012737  000001  001176    MOV     #1,$TMP1                ;RANGE OF PATTERN
 4409  013142  004737  042100            JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
 4410
 4411                            ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
 4412  013146  012702  001533            MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
 4413  013152  112722  000034            MOVB    #RMDC,(R2)+
 4414  013156  112722  000006            MOVB    #RMDA,(R2)+
 4415  013162  112722  000004            MOVB    #RMBA,(R2)+
 4416  013166  112722  000002            MOVB    #RMWC,(R2)+
 4417  013172  112722  000032            MOVB    #RMOF,(R2)+
 4418  013176  112722  000000            MOVB    #RMCS1,(R2)+
 4419  013202  112712  000200            MOVB    #200,(R2)               ;TERMINATE TABLE
 4420  013206                    30$:
 4421
 4422                            ;FORMAT THE DRIVE
 4423  013206  004737  043246            JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
 4424  013212  000404                    BR      40$                     ;GO TO 40$ IF NO ERROR
 4425  013214  000240                    NOP                     ;RETURN HERE IF ERROR
 4426  013216  104000                    ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
 4427  013220  000137  014016            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4428  013224                    40$:
 4429
 4430                            ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
 4431  013224  004737  042712            JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
 4432
 4433                            ;WAIT FOR THE FORMAT TO COMPLETE
 4434  013230  004737  043606            JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
 4435
 4436                            ;GO READ STATUS FOR FORMAT OPERATION
 4437  013234  004737  042776            JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 4438  013240  000404                    BR      50$                     ;GO TO 50$ IF NO ERROR
 4439  013242  000240                    NOP                     ;RETURN HERE IF ERROR
 4440  013244  104000                    ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 4441  013246  000137  014016            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4442  013252                    50$:
 4443
 4444                            ;VERIFY NO ERRORS DURING FORMAT
 4445  013252  004737  056276            JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 4446  013256  000405                    BR      60$                     ;GO TO 60$ IF NO ERROR
 4447  013260  000240                    NOP                     ;RETURN HERE IF ERROR
 4448  013262  104000                    ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 4449  013264  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4450  013266  000137  014016            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4451  013272                    60$:
 4452
 4453                            ;MOVE LOOP ADDRESSES TO NEXT OPERATION
 4454  013272  012737  013310  001122    MOV     #70$,$LPADR
 4455  013300  012737  013310  001124    MOV     #70$,$LPERR
 4456  013306  000410                    BR      80$                     ;SKIP TO WRITE OPERATION
```

```
                                ;;*****************************************************************
                                ;LOOP #2        WRITE,READ

013310                          70$:

                                ;PREPARE DEVICE FOR WRITE OPERATION
013310  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
013314  054130                          .WORD   054130  ;TASK DESCRIPTOR
013316  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
013320  000240                          NOP                     ;RETURN HERE IF ERROR
013322  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
013324  000137  014016                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
013330                          80$:

013330                          120$:

                                ;WRITE DATA TO THE DRIVE
013330  012737  106524  001402        MOV     #BUFONE+4,RMBAO         ;CHANGE MEMORY ADDRESS
013336  012737  177400  001400        MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
013344  012737  000061  001376        MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
013352  012702  001533                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
013356  112722  000006                MOVB    #RMDA,(R2)+
013362  112722  000034                MOVB    #RMDC,(R2)+
013366  112722  000032                MOVB    #RMOF,(R2)+
013372  112722  000004                MOVB    #RMBA,(R2)+
013376  112722  000002                MOVB    #RMWC,(R2)+
013402  112722  000000                MOVB    #RMCS1,(R2)+
013406  112722  000200                MOVB    #200,(R2)+              ;TERMINATE TABLE

013412  004737  043246                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
013416  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
013420  000240                          NOP                     ;RETURN HERE IF ERROR
013422  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
013424  000137  014016                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
013430                          130$:

                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
013430  004737  042712                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR WRITE COMMAND TO COMPLETE
013434  004737  043606                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR WRITE COMMAND
013440  004737  042776                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
013444  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
013446  000240                          NOP                     ;RETURN HERE IF ERROR
013450  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
013452  000137  014016                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
013456                          140$:

                                ;CHECK FOR ERRORS DURING WRITE OPERATION
013456  004737  043772                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
013462  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
```

```
4513  013464  000240                          NOP                             ;RETURN HERE IF ERROR
4514  013466  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
4515  013470  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4516  013472  000137  014016                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
4517  013476                          150$:
4518  013476  004737  056276                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
4519  013502  000405                          BR      160$                            ;GO TO 160$ IF NO ERROR
4520  013504  000240                          NOP                             ;RETURN HERE IF ERROR
4521  013506  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
4522  013510  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4523  013512  000137  014016                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
4524  013516                          160$:
4525
4526  013516                          170$:
4527  013516  004737  044624                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
4528  013522  000405                          BR      180$                            ;GO TO 180$ IF NO ERROR
4529  013524  000240                          NOP                             ;RETURN HERE IF ERROR
4530  013526  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
4531  013530  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4532  013532  000137  014016                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
4533  013536                          180$:
4534
4535                                  ;CHANGE LOOP ADDRESSES
4536  013536  012737  013554  001122          MOV     #190$,$LPADR
4537  013544  012737  013554  001124          MOV     #190$,$LPERR
4538  013552  000410                          BR      200$                            ;SKIP TO NEXT OPERATION
4539
4540                                  ;;*****************************************************************
4541                                  ;LOOP #3        READ
4542
4543  013554                          190$:
4544
4545                                  ;PREPARE DEVICE FOR READ OPERATION
4546  013554  004737  037260                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
4547  013560  054130                          .WORD   054130  ;TASK DESCRIPTOR
4548  013562  000404                          BR      200$                            ;GO TO 200$ IF NO ERROR
4549  013564  000240                          NOP                             ;RETURN HERE IF ERROR
4550  013566  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4551  013570  000137  014016                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
4552  013574                          200$:
4553
4554  013574                          240$:
4555
4556                                  ;READ DATA FROM DEVICE
4557  013574  012737  107530  001402          MOV     #BUFTWO+4,RMBAO         ;CHANGE MEMORY ADDRESS
4558  013602  012737  000071  001376          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
4559  013610  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
4560  013614  112722  000006                  MOVB    #RMDA,(R2)+
4561  013620  112722  000032                  MOVB    #RMOF,(R2)+
4562  013624  112722  000034                  MOVB    #RMDC,(R2)+
4563  013630  112722  000004                  MOVB    #RMBA,(R2)+
4564  013634  112722  000002                  MOVB    #RMWC,(R2)+
4565  013640  112722  000000                  MOVB    #RMCS1,(R2)+
4566  013644  112712  000200                  MOVB    #200,(R2)
4567
4568  013650  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
```

```
4569  013654  000404                T7      WRITE, READ ONES
4570  013656  000240                        BR      250$                    ;GO TO 250$ IF NO ERROR
4571  013660  104000                        NOP                     ;RETURN HERE IF ERROR
4572  013662  000137  014016              ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4573  013666                              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4574                                    250$:
4575                                    ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
4576  013666  004737  042712                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4577
4578                                    ;WAIT FOR READ OPERATION TO COMPLETE
4579  013672  004737  043606                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4580
4581                                    ;GO READ STATUS FOR READ OPERATION
4582  013676  004737  042776                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4583  013702  000404                        BR      260$                    ;GO TO 260$ IF NO ERROR
4584  013704  000240                        NOP                     ;RETURN HERE IF ERROR
4585  013706  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4586  013710  000137  014016                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4587  013714                              260$:
4588
4589                                    ;CHECK FOR ERRORS DURING READ OPERATION
4590  013714  004737  043772                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4591  013720  000405                        BR      270$                    ;GO TO 270$ IF NO ERROR
4592  013722  000240                        NOP                     ;RETURN HERE IF ERROR
4593  013724  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4594  013726  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4595  013730  000137  014016                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4596  013734                              270$:
4597  013734  004737  056276                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4598  013740  000405                        BR      280$                    ;GO TO 280$ IF NO ERROR
4599  013742  000240                        NOP                     ;RETURN HERE IF ERROR
4600  013744  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4601  013746  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4602  013750  000137  014016                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4603  013754                              280$:
4604
4605  013754                              290$:
4606  013754  004737  044624                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4607  013760  000405                        BR      300$                    ;GO TO 300$ IF NO ERROR
4608  013762  000240                        NOP                     ;RETURN HERE IF ERROR
4609  013764  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4610  013766  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4611  013770  000137  014016                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4612  013774                              300$:
4613  013774  004737  042344                JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
4614  014000  106524                        .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
4615  014002  107530                        .WORD   BUFTWO+4                ;STARTING ADDRESS OF READ BUFFER
4616  014004  000404                        BR      310$                    ;GO TO 310$ IF NO ERROR
4617  014006  000240                        NOP                     ;RETURN HERE IF ERROR
4618  014010  104000                        ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
4619  014012  000137  014016                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4620  014016                              310$:
4621
4622  014016                              350$:
4623                                    ;;*****************************************************************
4624                                    ;*TEST 10        WRITE, WRITE CHECK ONES
```

```
                                        ;;*****************************************************************
                                        T10:
                                                SCOPE                   ;SCOPE CALL
                                                NOP                     ;START OF TEST
014022  012706  001100                          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
014026  013700  001276                          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
014032  013701  001446                          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
014036  012737  000010  001226                  MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                        ;;*****************************************************************
                                        ;LOOP #1        FORMAT,WRITE,WRITE CHECK DATA

014044                                  10$:

                                        ;PREPARE THE DEVICE FOR FORMAT OPERATION
014044  004737  037260                          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
014050  054130                                  .WORD   054130  ;TASK DESCRIPTOR
014052  000404                                  BR      20$             ;GO TO 20$ IF NO ERROR
014054  000240                                  NOP                     ;RETURN HERE IF ERROR
014056  104000                                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
014060  000137  015010                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
014064                                  20$:

                                        ;LOAD PARAMETERS AND GENERATE DATA BUFFER
014064  012737  000000  001432                  MOV     #0,RMDC0                ;CYLINDER = 0
014072  012737  000000  001404                  MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
014100  012737  106520  001402                  MOV     #BUFONE,RMBA0           ;BUS ADDRESS
014106  012737  177376  001400                  MOV     #^C<2+256.>+1>,RMWC0            ;WORD COUNT = 1 SECTOR
014114  012737  010000  001430                  MOV     #FMT16,RMOF0            ;16 BIT FORMAT
014122  012737  000063  001376                  MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND

                                        ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE

014130  004737  040174                          JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
014134  000405                                  BR      25$                     ;GO TO 25$ IF NO ERROR
014136  104401  066704                          TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
014142  104000                                  ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
014144  000137  015010                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
014150                                  25$:
014150  012737  070016  001174                  MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
014156  012737  000001  001176                  MOV     #1,$TMP1                ;RANGE OF PATTERN
014164  004737  042100                          JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT

                                        ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
014170  012702  001533                          MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
014174  112722  000034                          MOVB    #RMDC,(R2)+
014200  112722  000006                          MOVB    #RMDA,(R2)+
014204  112722  000004                          MOVB    #RMBA,(R2)+
014210  112722  000002                          MOVB    #RMWC,(R2)+
014214  112722  000032                          MOVB    #RMOF,(R2)+
014220  112722  000000                          MOVB    #RMCS1,(R2)+
014224  112712  000200                          MOVB    #200,(R2)               ;TERMINATE TABLE
014230                                  30$:

                                        ;FORMAT THE DRIVE
014230  004737  043246                          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
```

```
 4681  014234  000404                        BR      40$                     ;GO TO 40$ IF NO ERROR
 4682  014236  000240                        NOP                     ;RETURN HERE IF ERROR
 4683  014240  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
 4684  014242  000137  015010                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4685  014246                        40$:
 4686
 4687                                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
 4688  014246  004737  042712                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
 4689
 4690                                ;WAIT FOR THE FORMAT TO COMPLETE
 4691  014252  004737  043606                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
 4692
 4693                                ;GO READ STATUS FOR FORMAT OPERATION
 4694  014256  004737  042776        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 4695  014262  000404                        BR      50$                     ;GO TO 50$ IF NO ERROR
 4696  014264  000240                        NOP                     ;RETURN HERE IF ERROR
 4697  014266  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 4698  014270  000137  015010                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4699  014274                        50$:
 4700
 4701                                ;VERIFY NO ERRORS DURING FORMAT
 4702  014274  004737  056276                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 4703  014300  000405                        BR      60$                     ;GO TO 60$ IF NO ERROR
 4704  014302  000240                        NOP                     ;RETURN HERE IF ERROR
 4705  014304  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 4706  014306  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4707  014310  000137  015010                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4708  014314                        60$:
 4709
 4710                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
 4711  014314  012737  014332  001122        MOV     #70$,$LPADR
 4712  014322  012737  014332  001124        MOV     #70$,$LPERR
 4713  014330  000410                        BR      80$                             ;SKIP TO WRITE OPERATION
 4714
 4715                                ;;*****************************************************************
 4716                                ;LOOP #2        WRITE,WRITE CHECK DATA
 4717
 4718  014332                        70$:
 4719
 4720
 4721                                ;PREPARE DEVICE FOR WRITE OPERATION
 4722  014332  004737  037260        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 4723  014336  054130                .WORD   054130  ;TASK DESCRIPTOR
 4724  014340  000404                BR      80$                     ;GO TO 80$ IF NO ERROR
 4725  014342  000240                NOP                     ;RETURN HERE IF ERROR
 4726  014344  104000                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 4727  014346  000137  015010        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4728  014352                        80$:
 4729
 4730
 4731  014352                        120$:
 4732
 4733                                ;WRITE DATA TO THE DRIVE
 4734  014352  012737  106524  001402        MOV     #BUFONE+4,RMBAO         ;CHANGE MEMORY ADDRESS
 4735  014360  012737  177400  001400        MOV     #<-C256.+1>,RMWCO       ;CHANGE WORD COUNT
 4736  014366  012737  000061  001376        MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
```

```
4737  014374  012702  001533          MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
4738  014400  112722  000006          MOVB    #RMDA,(R2)+
4739  014404  112722  000034          MOVB    #RMDC,(R2)+
4740  014410  112722  000032          MOVB    #RMOF,(R2)+
4741  014414  112722  000004          MOVB    #RMBA,(R2)+
4742  014420  112722  000002          MOVB    #RMWC,(R2)+
4743  014424  112722  000000          MOVB    #RMCS1,(R2)+
4744  014430  112722  000200          MOVB    #200,(R2)+              ;TERMINATE TABLE
4745
4746  014434  004737  043246          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4747  014440  000404                  BR      130$                    ;GO TO 130$ IF NO ERROR
4748  014442  000240                  NOP                     ;RETURN HERE IF ERROR
4749  014444  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4750  014446  000137  015010          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4751  014452                  130$:
4752
4753                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
4754  014452  004737  042712          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4755
4756                          ;WAIT FOR WRITE COMMAND TO COMPLETE
4757  014456  004737  043606          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4758
4759                          ;GO READ STATUS FOR WRITE COMMAND
4760  014462  004737  042776          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4761  014466  000404                  BR      140$                    ;GO TO 140$ IF NO ERROR
4762  014470  000240                  NOP                     ;RETURN HERE IF ERROR
4763  014472  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4764  014474  000137  015010          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4765  014500                  140$:
4766
4767                          ;CHECK FOR ERRORS DURING WRITE OPERATION
4768  014500  004737  043772          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4769  014504  000405                  BR      150$                    ;GO TO 150$ IF NO ERROR
4770  014506  000240                  NOP                     ;RETURN HERE IF ERROR
4771  014510  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4772  014512  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4773  014514  000137  015010          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4774  014520                  150$:
4775  014520  004737  056276          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4776  014524  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
4777  014526  000240                  NOP                     ;RETURN HERE IF ERROR
4778  014530  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4779  014532  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4780  014534  000137  015010          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4781  014540                  160$:
4782
4783  014540                  170$:
4784  014540  004737  044624          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4785  014544  000405                  BR      180$                    ;GO TO 180$ IF NO ERROR
4786  014546  000240                  NOP                     ;RETURN HERE IF ERROR
4787  014550  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4788  014552  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4789  014554  000137  015010          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4790  014560                  180$:
4791
4792                          ;CHANGE LOOP ADDRESSES
```

```
 4793  014560  012737  014576  001122          MOV     #190$,$LPADR
 4794  014566  012737  014576  001124          MOV     #190$,$LPERR
 4795  014574  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
 4796
 4797                                  ;;*****************************************************************
 4798                                  ;LOOP #3        WRITE CHECK DATA
 4799
 4800  014576                          190$:
 4801
 4802                                  ;PREPARE DEVICE FOR WRITE CHECK DATA OPERATION
 4803  014576  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 4804  014602  054130                          .WORD   054130  ;TASK DESCRIPTOR
 4805  014604  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
 4806  014606  000240                          NOP                     ;RETURN HERE IF ERROR
 4807  014610  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 4808  014612  000137  015010                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4809  014616                          200$:
 4810
 4811  014616                          240$:
 4812
 4813                                  ;WRITE CHECK DATA DATA FROM DEVICE
 4814  014616  012737  000051  001376          MOV     #WCD!GO,RMCS10          ;WRITE CHECK DATA DATA COMMAND
 4815  014624  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
 4816  014630  112722  000006                  MOVB    #RMDA,(R2)+
 4817  014634  112722  000032                  MOVB    #RMOF,(R2)+
 4818  014640  112722  000034                  MOVB    #RMDC,(R2)+
 4819  014644  112722  000004                  MOVB    #RMBA,(R2)+
 4820  014650  112722  000002                  MOVB    #RMWC,(R2)+
 4821  014654  112722  000000                  MOVB    #RMCS1,(R2)+
 4822  014660  112712  000200                  MOVB    #200,(R2)
 4823
 4824  014664  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
 4825  014670  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
 4826  014672  000240                          NOP                     ;RETURN HERE IF ERROR
 4827  014674  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
 4828  014676  000137  015010                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4829  014702                          250$:
 4830
 4831                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
 4832  014702  004737  042712                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
 4833
 4834                                  ;WAIT FOR WRITE CHECK DATA OPERATION TO COMPLETE
 4835  014706  004737  043606                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
 4836
 4837                                  ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
 4838  014712  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 4839  014716  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
 4840  014720  000240                          NOP                     ;RETURN HERE IF ERROR
 4841  014722  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 4842  014724  000137  015010                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 4843  014730                          260$:
 4844
 4845                                  ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
 4846  014730  004737  043772                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 4847  014734  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
 4848  014736  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
4849  014740  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4850  014742  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4851  014744  000137  015010                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4852  014750                        270$:
4853  014750  004737  056276                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4854  014754  000405                        BR      280$                    ;GO TO 280$ IF NO ERROR
4855  014756  000240                        NOP                     ;RETURN HERE IF ERROR
4856  014760  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4857  014762  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4858  014764  000137  015010                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4859  014770                        280$:
4860
4861  014770                        290$:
4862  014770  004737  044624                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4863  014774  000405                        BR      300$                    ;GO TO 300$ IF NO ERROR
4864  014776  000240                        NOP                     ;RETURN HERE IF ERROR
4865  015000  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4866  015002  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4867  015004  000137  015010                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4868  015010                        300$:
4869
4870  015010                        350$:
4871                                ;;*****************************************************************
4872                                ;*TEST 11       WRITE, WRITE CHECK ONES W/ WCE ERROR
4873                                ;;*****************************************************************
4874  015010                        TST11:
4875  015010  000004                        SCOPE                   ;SCOPE CALL
4876  015012  000240                        NOP                     ;START OF TEST
4877  015014  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4878  015020  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4879  015024  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4880  015030  012737  000011  001226        MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4881
4882                                ;;*****************************************************************
4883                                ;LOOP #1        FORMAT,WRITE,WRITE CHECK DATA
4884
4885  015036                        10$:
4886
4887                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
4888  015036  004737  037260                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4889  015042  054130                        .WORD   054130  ;TASK DESCRIPTOR
4890  015044  000404                        BR      20$                     ;GO TO 20$ IF NO ERROR
4891  015046  000240                        NOP                     ;RETURN HERE IF ERROR
4892  015050  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4893  015052  000137  016230                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4894  015056                        20$:
4895
4896                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
4897  015056  012737  000000  001432        MOV     #0,RMDC0                ;CYLINDER = 0
4898  015064  012737  000000  001404        MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
4899  015072  012737  106520  001402        MOV     #BUFONE,RMBA0           ;BUS ADDRESS
4900  015100  012737  177376  001400        MOV     #<^C<2+256.>+1>,RMWC0           ;WORD COUNT = 1 SECTOR
4901  015106  012737  010000  001430        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
4902  015114  012737  000063  001376        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
4903
4904                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
```

```
4905
4906  015122  004737  040174          JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
4907  015126  000405                  BR      25$                     ;GO TO 25$ IF NO ERROR
4908  015130  104401  066704          TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
4909  015134  104000                  ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
4910  015136  000137  016230          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4911  015142                  25$:
4912  015142  012737  070016  001174          MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
4913  015150  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
4914  015156  004737  042100          JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
4915
4916                          ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
4917  015162  012702  001533          MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
4918  015166  112722  000034          MOVB    #RMDC,(R2)+
4919  015172  112722  000006          MOVB    #RMDA,(R2)+
4920  015176  112722  000004          MOVB    #RMBA,(R2)+
4921  015202  112722  000002          MOVB    #RMWC,(R2)+
4922  015206  112722  000032          MOVB    #RMOF,(R2)+
4923  015212  112722  000000          MOVB    #RMCS1,(R2)+
4924  015216  112712  000200          MOVB    #200,(R2)               ;TERMINATE TABLE
4925  015222                  30$:
4926
4927                          ;FORMAT THE DRIVE
4928  015222  004737  043246          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4929  015226  000404                  BR      40$                     ;GO TO 40$ IF NO ERROR
4930  015230  000240                  NOP                     ;RETURN HERE IF ERROR
4931  015232  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4932  015234  000137  016230          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4933  015240                  40$:
4934
4935                          ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
4936  015240  004737  042712          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4937
4938                          ;WAIT FOR THE FORMAT TO COMPLETE
4939  015244  004737  043606          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4940
4941                          ;GO READ STATUS FOR FORMAT OPERATION
4942  015250  004737  042776          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4943  015254  000404                  BR      50$                     ;GO TO 50$ IF NO ERROR
4944  015256  000240                  NOP                     ;RETURN HERE IF ERROR
4945  015260  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4946  015262  000137  016230          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4947  015266                  50$:
4948
4949                          ;VERIFY NO ERRORS DURING FORMAT
4950  015266  004737  056276          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4951  015272  000405                  BR      60$                     ;GO TO 60$ IF NO ERROR
4952  015274  000240                  NOP                     ;RETURN HERE IF ERROR
4953  015276  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4954  015300  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4955  015302  000137  016230          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4956  015306                  60$:
4957
4958                          ;MOVE LOOP ADDRESSES TO NEXT OPERATION
4959  015306  012737  015324  001122          MOV     #70$,$LPADR
4960  015314  012737  015324  001124          MOV     #70$,$LPERR
```

```
                                        BR      80$                     ;SKIP TO WRITE OPERATION

                                ;;*****************************************************************
                                ;LOOP #2        WRITE,WRITE CHECK DATA

                                70$:

                                ;PREPARE DEVICE FOR WRITE OPERATION
                                        JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
                                        .WORD   054130  ;TASK DESCRIPTOR
                                        BR      80$                     ;GO TO 80$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                80$:

                                120$:

                                ;WRITE DATA TO THE DRIVE
                                        MOV     #<^C256.+1>,RMWC0       ;CHANGE WORD COUNT
                                        MOV     #BUFONE+4,RMBA0         ;CHANGE MEMORY ADDRESS
                                        MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
                                        MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
                                        MOVB    #RMDA,(R2)+
                                        MOVB    #RMDC,(R2)+
                                        MOVB    #RMOF,(R2)+
                                        MOVB    #RMBA,(R2)+
                                        MOVB    #RMWC,(R2)+
                                        MOVB    #RMCS1,(R2)+
                                        MOVB    #200,(R2)+              ;TERMINATE TABLE

                                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
                                        BR      130$                    ;GO TO 130$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                130$:

                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
                                        JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR WRITE COMMAND TO COMPLETE
                                        JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR WRITE COMMAND
                                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
                                        BR      140$                    ;GO TO 140$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                140$:

                                ;CHECK FOR ERRORS DURING WRITE OPERATION
                                        JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
```

```
5017  015476  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
5018  015500  000240                        NOP                     ;RETURN HERE IF ERROR
5019  015502  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5020  015504  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5021  015506  000137  016230                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5022  015512                          150$:
5023  015512  004737  056276                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5024  015516  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
5025  015520  000240                        NOP                     ;RETURN HERE IF ERROR
5026  015522  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5027  015524  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5028  015526  000137  016230                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5029  015532                          160$:
5030
5031  015532                          170$:
5032  015532  004737  044624                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5033  015536  000405                        BR      180$                    ;GO TO 180$ IF NO ERROR
5034  015540  000240                        NOP                     ;RETURN HERE IF ERROR
5035  015542  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5036  015544  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5037  015546  000137  016230                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5038  015552                          180$:
5039
5040                                  ;CHANGE LOOP ADDRESSES
5041  015552  012737  015574  001122        MOV     #190$,$LPADR
5042  015560  012737  015574  001124        MOV     #190$,$LPERR
5043  015566  012737  015600  001210        MOV     #195$,$ESCAPE   ;;ESCAPE TO 195$ ON ERROR
5044
5045                                  ;;*****************************************************************
5046                                  ;LOOP #3        WRITE CHECK DATA
5047
5048  015574                          190$:
5049
5050  015574  012703  000001                MOV     #1,R3                   ;R3+WCE BIT POSITION
5051  015600  040337  107522          195$: BIC     R3,BUFTWO-2             ;CHANGE LAST WORD OF BUFFER
5052
5053                                  ;PREPARE DEVICE FOR WRITE CHECK DATA OPERATION
5054  015604  004737  037260                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5055  015610  054130                        .WORD   054130  ;TASK DESCRIPTOR
5056  015612  000404                        BR      200$                    ;GO TO 200$ IF NO ERROR
5057  015614  000240                        NOP                     ;RETURN HERE IF ERROR
5058  015616  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5059  015620  000137  016230                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5060  015624                          200$:
5061
5062  015624                          240$:
5063
5064                                  ;READ DATA FROM DEVICE
5065  015624  012737  000051  001376        MOV     #WCD!GO,RMCS10          ;WRITE CHECK DATA DATA COMMAND
5066  015632  012702  001533                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5067  015636  112722  000006                MOVB    #RMDA,(R2)+
5068  015642  112722  000032                MOVB    #RMOF,(R2)+
5069  015646  112722  000034                MOVB    #RMDC,(R2)+
5070  015652  112722  000004                MOVB    #RMBA,(R2)+
5071  015656  112722  000002                MOVB    #RMWC,(R2)+
5072  015662  112722  000000                MOVB    #RMCS1,(R2)+
```

```
5073  015666  112712  000200                    MOVB    #200,(R2)
5074
5075  015672  004737  043246                    JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5076  015676  000404                            BR      250$                    ;GO TO 250$ IF NO ERROR
5077  015700  000240                            NOP                     ;RETURN HERE IF ERROR
5078  015702  104000                            ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5079  015704  000137  016230                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5080  015710                          250$:
5081
5082                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5083  015710  004737  042712                    JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5084
5085                                  ;WAIT FOR WRITE CHECK DATA OPERATION TO COMPLETE
5086  015714  004737  043606                    JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5087
5088                                  ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
5089  015720  004737  042776                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5090  015724  000404                            BR      260$                    ;GO TO 260$ IF NO ERROR
5091  015726  000240                            NOP                     ;RETURN HERE IF ERROR
5092  015730  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5093  015732  000137  016230                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5094  015736                          260$:
5095
5096                                  ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
5097  015736  004737  043772                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5098  015742  000405                            BR      270$                    ;GO TO 270$ IF NO ERROR
5099  015744  000240                            NOP                     ;RETURN HERE IF ERROR
5100  015746  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5101  015750  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5102  015752  000137  016230                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5103  015756                          270$:
5104  015756  032737  040000  001336            BIT     #WCE,RMCS2I             ;WAS "WCE" DETECTED??
5105  015764  001030                            BNE     285$                    ;YES!!
5106  015766  004737  056276                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5107  015772  000405                            BR      280$                    ;GO TO 280$ IF NO ERROR
5108  015774  000240                            NOP                     ;RETURN HERE IF ERROR
5109  015776  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5110  016000  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5111  016002  000137  016230                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5112  016006                          280$:
5113  016006  013737  001336  001140            MOV     RMCS2I,$GDDAT           ;EXPECTED STATUS
5114  016014  052737  040000  001140            BIS     #WCE,$GDDAT
5115  016022  013737  001336  001142            MOV     RMCS2I,$BDDAT           ;RECEIVED STATUS
5116  016030  010337  001174                    MOV     R3,$TMP0                ;FAILING BIT POSITION
5117  016034  012737  107522  001176            MOV     #BUFTWO-2,$TMP1         ;FAILING ADDRESS
5118  016042  104337                            ERROR   337                     ;WCE NOT DETECTED
5119  016044  000471                            BR      350$
5120
5121  016046                          285$:
5122  016046  112737  000022  001504            MOVB    #RMDB,GETINX    ;SETUP GET INDEX TABLE
5123  016054  112737  000200  001505            MOVB    #200,GETINX+1
5124  016062  012737  016230  001350            MOV     #350$,RMDBI     ;SET THE INPUT BUFFER
5125  016070  004737  042776                    JSR     PC,GET          ;GO GET THE CONTENTS OF THE DATA BUFFER
5126  016074  000402                            BR      290$            ;GO TO 290$ IF NO ERROR
5127  016076  000240                            NOP                     ;RETURN HERE IF ERROR
5128  016100  104000                            ERROR                   ;ERROR DEFINED BY GET SUBROUTINE
```

```
5129
5130  016102                          290$:
5131  016102  013737  001350  001142          MOV     RMDBI,$BDDAT            ;RECEIVED DATA
5132  016110  013737  107522  001140          MOV     BUFTWO-2,$GDDAT         ;EXPECTED DATA
5133  016116  050337  001140                  BIS     R3,$GDDAT
5134  016122  012737  107522  001134          MOV     #BUFTWO-2,$GDADR        ;EXPECTED ADDRESS
5135  016130  013737  001332  001136          MOV     RMBAI,$BDADR            ;RECEIVED ADDRESS
5136  016136  162737  000002  001136          SUB     #2,$BDADR               ;CORRECT MEMORY ADDRESS
5137  016144  023737  001134  001136          CMP     $GDADR,$BDADR           ;ADDRESSES OK??
5138  016152  001402                          BEQ     295$                    ;YES!!
5139  016154  104340                          ERROR   340                     ;WCE AT UNEXPECTED ADDRESS
5140  016156  000424                          BR      350$
5141  016160  023737  001140  001142  295$:   CMP     $GDDAT,$BDDAT           ;DATA OK??
5142  016166  001402                          BEQ     296$                    ;YES!!
5143  016170  104341                          ERROR   341                     ;UNEXPECTED WCE DATA
5144  016172  000416                          BR      350$
5145  016174                          296$:
5146
5147  016174  004737  044624                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5148  016200  000405                          BR      300$                    ;GO TO 300$ IF NO ERROR
5149  016202  000240                          NOP                     ;RETURN HERE IF ERROR
5150  016204  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5151  016206  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5152  016210  000137  016230                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5153  016214                          300$:
5154
5155  016214  050337  107522                  BIS     R3,BUFTWO-2             ;RESTORE DATA PATTERN
5156  016220  006303                          ASL     R3                      ;SHIFT TO NEXT BIT
5157  016222  001402                          BEQ     350$                    ;EXIT IF DONE
5158  016224  000137  015600                  JMP     195$                    ;REPEAT TEST FOR NEXT DATA BIT
5159  016230                          350$:
5160  016230  012737  000000  001210          MOV     #0,$ESCAPE      ;;ESCAPE TO 0 ON ERROR
5161  016236  012737  015036  001122          MOV     #10$,$LPADR             ;CHANGE LOOP TO START OF TEST
5162  016244  012737  015036  001124          MOV     #10$,$LPERR
5163                          ;;*****************************************************************
5164                          ;*TEST 12        WRITE, WRITE CHECK MULTIPLE SECTORS
5165                          ;;*****************************************************************
5166  016252                  TST12:
5167  016252  000004                          SCOPE                   ;SCOPE CALL
5168  016254  000240                          NOP                     ;START OF TEST
5169  016256  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5170  016262  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5171  016266  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5172  016272  012737  000012  001226          MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5173
5174                          ;;*****************************************************************
5175                          ;LOOP #1        FORMAT,WRITE,READ
5176
5177  016300                  10$:
5178
5179                          ;PREPARE THE DEVICE FOR FORMAT OPERATION
5180  016300  004737  037260          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5181  016304  054130          .WORD   054130  ;TASK DESCRIPTOR
5182  016306  000404                  BR      20$                     ;GO TO 20$ IF NO ERROR
5183  016310  000240                  NOP                     ;RETURN HERE IF ERROR
5184  016312  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
5185  016314  000137  017272                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5186  016320                                20$:
5187
5188                                        ;LOAD PARAMETERS AND GENERATE DATA BUFFER
5189  016320  012737  000000  001432            MOV     #0,RMDC0                ;CYLINDER = 0
5190  016326  012737  000000  001404            MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
5191  016334  012737  106520  001402            MOV     #BUFONE,RMBA0           ;BUS ADDRESS
5192  016342  012737  177240  001400            MOV     #^C<2*<2+256>>+1>,RMWC0         ;WORD COUNT
5193  016350  012737  010000  001430            MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5194  016356  012737  000063  001376            MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
5195
5196                                        ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
5197
5198  016364  004737  040174                    JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
5199  016370  000405                            BR      25$                     ;GO TO 25$ IF NO ERROR
5200  016372  104401  066704                    TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
5201  016376  104000                            ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
5202  016400  000137  017272                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5203  016404                                25$:
5204  016404  012737  070060  001174            MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
5205  016412  012737  000001  001176            MOV     #1,$TMP1                ;RANGE OF PATTERN
5206  016420  004737  042100                    JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
5207
5208                                        ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
5209  016424  012702  001533                    MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
5210  016430  112722  000034                    MOVB    #RMDC,(R2)+
5211  016434  112722  000006                    MOVB    #RMDA,(R2)+
5212  016440  112722  000004                    MOVB    #RMBA,(R2)+
5213  016444  112722  000002                    MOVB    #RMWC,(R2)+
5214  016450  112722  000032                    MOVB    #RMOF,(R2)+
5215  016454  112722  000000                    MOVB    #RMCS1,(R2)+
5216  016460  112712  000200                    MOVB    #200,(R2)               ;TERMINATE TABLE
5217  016464                                30$:
5218
5219                                        ;FORMAT THE DRIVE
5220  016464  004737  043246                    JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5221  016470  000404                            BR      40$                     ;GO TO 40$ IF NO ERROR
5222  016472  000240                            NOP                     ;RETURN HERE IF ERROR
5223  016474  104000                            ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5224  016476  000137  017272                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5225  016502                                40$:
5226
5227                                        ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
5228  016502  004737  042712                    JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5229
5230                                        ;WAIT FOR THE FORMAT TO COMPLETE
5231  016506  004737  043606                    JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5232
5233                                        ;GO READ STATUS FOR FORMAT OPERATION
5234  016512  004737  042776                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5235  016516  000404                            BR      50$                     ;GO TO 50$ IF NO ERROR
5236  016520  000240                            NOP                     ;RETURN HERE IF ERROR
5237  016522  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5238  016524  000137  017272                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5239  016530                                50$:
5240
```

```
5241
5242  016530  004737  056276          ;VERIFY NO ERRORS DURING FORMAT
5243  016534  000405                          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5244  016536  000240                          BR      60$                     ;GO TO 60$ IF NO ERROR
5245  016540  104000                          NOP                     ;RETURN HERE IF ERROR
5246  016542  004736                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5247  016544  000137  017272                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5248  016550                                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5249                                  60$:
5250                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
5251  016550  012737  016630  001122          MOV     #70$,$LPADR
5252  016556  012737  016630  001124          MOV     #70$,$LPERR
5253                                  ;REGENERATE DATA BUFFER
5254  016564  012737  177000  001400          MOV     #<^C<2*256.>+1>,RMWC0   ;CHANGE WORD COUNT
5255  016572  012737  106520  001402          MOV     #BUFONE,RMBA0           ;CHANGE BUS ADDRESS
5256  016600  012737  000060  001376          MOV     #WD,RMCS10              ;WRITE DATA
5257  016606  012737  070016  001174          MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
5258  016614  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
5259  016622  004737  042100                JSR     PC,GENBUF
5260  016626  000410                          BR      80$                     ;SKIP TO WRITE OPERATION
5261
5262                                  ;;*****************************************************************
5263                                  ;LOOP #2        WRITE,READ
5264
5265  016630                          70$:
5266
5267
5268                                  ;PREPARE DEVICE FOR WRITE OPERATION
5269  016630  004737  037260                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5270  016634  054130                        .WORD   054130  ;TASK DESCRIPTOR
5271  016636  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
5272  016640  000240                        NOP                     ;RETURN HERE IF ERROR
5273  016642  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5274  016644  000137  017272                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5275  016650                          80$:
5276
5277
5278  016650                          120$:
5279
5280                                  ;WRITE DATA TO THE DRIVE
5281  016650  012737  000061  001376          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
5282  016656  012702  001533                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5283  016662  112722  000006                MOVB    #RMDA,(R2)+
5284  016666  112722  000034                MOVB    #RMDC,(R2)+
5285  016672  112722  000032                MOVB    #RMOF,(R2)+
5286  016676  112722  000004                MOVB    #RMBA,(R2)+
5287  016702  112722  000002                MOVB    #RMWC,(R2)+
5288  016706  112722  000000                MOVB    #RMCS1,(R2)+
5289  016712  112722  000200                MOVB    #200,(R2)+              ;TERMINATE TABLE
5290
5291  016716  004737  043246                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5292  016722  000404                        BR      130$                    ;GO TO 130$ IF NO ERROR
5293  016724  000240                        NOP                     ;RETURN HERE IF ERROR
5294  016726  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5295  016730  000137  017272                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5296  016734                          130$:
```

```
5297
5298                                    ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5299  016734  004737  042712                    JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5300
5301                                    ;WAIT FOR WRITE COMMAND TO COMPLETE
5302  016740  004737  043606                    JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5303
5304                                    ;GO READ STATUS FOR WRITE COMMAND
5305  016744  004737  042776                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5306  016750  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
5307  016752  000240                          NOP                     ;RETURN HERE IF ERROR
5308  016754  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5309  016756  000137  017272                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5310  016762                          140$:
5311
5312                                    ;CHECK FOR ERRORS DURING WRITE OPERATION
5313  016762  004737  043772                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5314  016766  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
5315  016770  000240                          NOP                     ;RETURN HERE IF ERROR
5316  016772  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5317  016774  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5318  016776  000137  017272                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5319  017002                          150$:
5320  017002  004737  056276                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5321  017006  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
5322  017010  000240                          NOP                     ;RETURN HERE IF ERROR
5323  017012  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5324  017014  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5325  017016  000137  017272                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5326  017022                          160$:
5327
5328  017022                          170$:
5329  017022  004737  044624                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5330  017026  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
5331  017030  000240                          NOP                     ;RETURN HERE IF ERROR
5332  017032  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5333  017034  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5334  017036  000137  017272                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5335  017042                          180$:
5336
5337                                    ;CHANGE LOOP ADDRESSES
5338  017042  012737  017060  001122          MOV     #190$,$LPADR
5339  017050  012737  017060  001124          MOV     #190$,$LPERR
5340  017056  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
5341
5342                                    ;;*****************************************************************
5343                                    ;LOOP #3        READ
5344
5345  017060                          190$:
5346
5347                                    ;PREPARE DEVICE FOR READ OPERATION
5348  017060  004737  037260                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5349  017064  054130                          .WORD   054130  ;TASK DESCRIPTOR
5350  017066  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
5351  017070  000240                          NOP                     ;RETURN HERE IF ERROR
5352  017072  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
5353  017074  000137  017272                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5354  017100                                200$:
5355
5356  017100                                240$:
5357
5358                                        ;READ DATA FROM DEVICE
5359  017100  012737  000051  001376              MOV     #WCD!GO,RMCS10          ;READ DATA COMMAND
5360  017106  012702  001533                    MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5361  017112  112722  000006                    MOVB    #RMDA,(R2)+
5362  017116  112722  000032                    MOVB    #RMOF,(R2)+
5363  017122  112722  000034                    MOVB    #RMDC,(R2)+
5364  017126  112722  000004                    MOVB    #RMBA,(R2)+
5365  017132  112722  000002                    MOVB    #RMWC,(R2)+
5366  017136  112722  000000                    MOVB    #RMCS1,(R2)+
5367  017142  112712  000200                    MOVB    #200,(R2)
5368
5369  017146  004737  043246                    JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5370  017152  000404                            BR      250$                    ;GO TO 250$ IF NO ERROR
5371  017154  000240                            NOP                     ;RETURN HERE IF ERROR
5372  017156  104000                            ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5373  017160  000137  017272                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5374  017164                                250$:
5375
5376                                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5377  017164  004737  042712                    JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5378
5379                                        ;WAIT FOR READ OPERATION TO COMPLETE
5380  017170  004737  043606                    JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5381
5382                                        ;GO READ STATUS FOR READ OPERATION
5383  017174  004737  042776                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5384  017200  000404                            BR      260$                    ;GO TO 260$ IF NO ERROR
5385  017202  000240                            NOP                     ;RETURN HERE IF ERROR
5386  017204  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5387  017206  000137  017272                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5388  017212                                260$:
5389
5390                                        ;CHECK FOR ERRORS DURING READ OPERATION
5391  017212  004737  043772                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5392  017216  000405                            BR      270$                    ;GO TO 270$ IF NO ERROR
5393  017220  000240                            NOP                     ;RETURN HERE IF ERROR
5394  017222  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5395  017224  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5396  017226  000137  017272                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5397  017232                                270$:
5398  017232  004737  056276                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5399  017236  000405                            BR      280$                    ;GO TO 280$ IF NO ERROR
5400  017240  000240                            NOP                     ;RETURN HERE IF ERROR
5401  017242  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5402  017244  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5403  017246  000137  017272                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5404  017252                                280$:
5405
5406  017252                                290$:
5407  017252  004737  044624                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5408  017256  000405                            BR      300$                    ;GO TO 300$ IF NO ERROR
```

```
5409  017260  000240                          NOP                     ;RETURN HERE IF ERROR
5410  017262  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5411  017264  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5412  017266  000137  017272                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5413  017272                          300$:
5414
5415  017272                          350$:
5416                                  ;;*****************************************************************
5417                                  ;*TEST 13       WRITE, READ W/ IMPLIED SEEK
5418                                  ;;*****************************************************************
5419  017272                          TST13:
5420  017272  000004                          SCOPE                   ;SCOPE CALL
5421  017274  000240                          NOP                     ;START OF TEST
5422  017276  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5423  017302  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5424  017306  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5425  017312  012737  000013  001226          MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5426
5427                                  ;;*****************************************************************
5428                                  ;LOOP #1        FORMAT,WRITE,READ
5429
5430  017320                          10$:
5431
5432                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
5433  017320  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5434  017324  054130                          .WORD   054130  ;TASK DESCRIPTOR
5435  017326  000404                          BR      20$                     ;GO TO 20$ IF NO ERROR
5436  017330  000240                          NOP                     ;RETURN HERE IF ERROR
5437  017332  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5438  017334  000137  020614                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5439  017340                          20$:
5440
5441                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
5442  017340  012737  000000  001432          MOV     #0,RMDC0                ;CYLINDER = 0
5443  017346  012737  000000  001404          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
5444  017354  012737  106520  001402          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
5445  017362  012737  177376  001400          MOV     #<-<2+256.>+1>,RMWC0    ;WORD COUNT
5446  017370  012737  010000  001430          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5447  017376  012737  000063  001376          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
5448
5449                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
5450
5451  017404  004737  040174                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
5452  017410  000405                          BR      25$                     ;GO TO 25$ IF NO ERROR
5453  017412  104401  066704                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
5454  017416  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
5455  017420  000137  020614                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5456  017424
5457  017424  012737  070016  001174  25$:    MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
5458  017432  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
5459  017440  004737  042100                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
5460
5461                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
5462  017444  012702  001533                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
5463  017450  112722  000034                  MOVB    #RMDC,(R2)+
5464  017454  112722  000006                  MOVB    #RMDA,(R2)+
```

```
5465  017460  112722  000004                  MOVB    #RMBA,(R2)+
5466  017464  112722  000002                  MOVB    #RMWC,(R2)+
5467  017470  112722  000032                  MOVB    #RMOF,(R2)+
5468  017474  112722  000000                  MOVB    #RMCS1,(R2)+
5469  017500  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
5470  017504                          30$:
5471
5472                                  ;FORMAT THE DRIVE
5473  017504  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5474  017510  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
5475  017512  000240                          NOP                     ;RETURN HERE IF ERROR
5476  017514  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5477  017516  000137  020614                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5478  017522                          40$:
5479
5480                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
5481  017522  004737  042712                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
5482
5483                                  ;WAIT FOR THE FORMAT TO COMPLETE
5484  017526  004737  043606                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
5485
5486                                  ;GO READ STATUS FOR FORMAT OPERATION
5487  017532  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5488  017536  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
5489  017540  000240                          NOP                     ;RETURN HERE IF ERROR
5490  017542  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5491  017544  000137  020614                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5492  017550                          50$:
5493
5494                                  ;VERIFY NO ERRORS DURING FORMAT
5495  017550  004737  056276                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
5496  017554  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
5497  017556  000240                          NOP                     ;RETURN HERE IF ERROR
5498  017560  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5499  017562  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5500  017564  000137  020614                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5501  017570                          60$:
5502
5503                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
5504  017570  012737  017630  001122          MOV     #70$,$LPADR
5505  017576  012737  017630  001124          MOV     #70$,$LPERR
5506  017604  013737  001432  020616          MOV     RMDCO,360$
5507  017612  012737  017622  001210          MOV     #65$,$ESCAPE    ;;ESCAPE TO 65$ ON ERROR
5508  017620  000413                          BR      80$                     ;SKIP TO WRITE OPERATION
5509
5510  017622  013737  020616  001432  65$:    MOV     360$,RMDCO              ;RESTORE CYLINDER
5511                                  ;;*****************************************************************
5512                                  ;LOOP #2        WRITE,READ
5513
5514  017630                          70$:
5515
5516
5517                                  ;PREPARE DEVICE FOR WRITE OPERATION
5518  017630  004737  037260                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
5519  017634  054130                          .WORD   054130  ;TASK DESCRIPTOR
5520  017636  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
```

```
5521  017640  000240                        NOP                     ;RETURN HERE IF ERROR
5522  017642  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5523  017644  000137  020614                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5524  017650                        80$:
5525
5526  017650  013737  001432  020616        MOV     RMDC0,360$              ;SAVE CYLINER
5527  017656  012737  001466  001432        MOV     #822.,RMDC0             ;SEEK TO LAST CYLINER
5528  017664  012737  000005  001376        MOV     #SEEK!GO,RMCS10         ;WRITE SEEK COMMAND
5529  017672  004737  043246                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5530  017676  000404                        BR      90$                     ;GO TO 90$ IF NO ERROR
5531  017700  000240                        NOP                     ;RETURN HERE IF ERROR
5532  017702  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5533  017704  000137  020614                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5534  017710                        90$:
5535
5536                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5537  017710  004737  042712                JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
5538  017714  004737  043606                JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
5539  017720  004737  042776                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5540  017724  000404                        BR      100$                    ;GO TO 100$ IF NO ERROR
5541  017726  000240                        NOP                     ;RETURN HERE IF ERROR
5542  017730  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5543  017732  000137  020614                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5544  017736                        100$:
5545  017736  004737  050700                JSR     PC,SEKSTS               ;GO VERIFY RESULTS OF SEEK OPERATION
5546  017742  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
5547  017744  000240                        NOP                     ;RETURN HERE IF ERROR
5548  017746  104000                        ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5549  017750  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5550  017752  000137  020614                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5551  017756  013737  020616  001432  110$:  MOV     360$,RMDC0              ;RESTORE CYLINDER
5552  017764  012737  000000  001210        MOV     #0,$ESCAPE              ;;ESCAPE TO 0 ON ERROR
5553
5554  017772                        120$:
5555
5556                                ;WRITE DATA TO THE DRIVE
5557  017772  012737  177400  001400        MOV     #<-C256.+1>,RMWC0       ;CHANGE WORD COUNT
5558  020000  012737  106524  001402        MOV     #BUF0NE+4,RMBA0         ;CHANGE BUS ADDRESS
5559  020006  012737  000061  001376        MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
5560  020014  012702  001533                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5561  020020  112722  000006                MOVB    #RMDA,(R2)+
5562  020024  112722  000034                MOVB    #RMDC,(R2)+
5563  020030  112722  000032                MOVB    #RMOF,(R2)+
5564  020034  112722  000004                MOVB    #RMBA,(R2)+
5565  020040  112722  000002                MOVB    #RMWC,(R2)+
5566  020044  112722  000000                MOVB    #RMCS1,(R2)+
5567  020050  112722  000200                MOVB    #200,(R2)+              ;TERMINATE TABLE
5568
5569  020054  004737  043246                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5570  020060  000404                        BR      130$                    ;GO TO 130$ IF NO ERROR
5571  020062  000240                        NOP                     ;RETURN HERE IF ERROR
5572  020064  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5573  020066  000137  020614                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5574  020072                        130$:
5575
5576                                ;WAIT FOR WRITE COMMAND TO COMPLETE
```

```
                                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR WRITE COMMAND
                                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
                                        BR      140$                    ;GO TO 140$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                140$:

                                ;CHECK FOR ERRORS DURING WRITE OPERATION
                                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                                        BR      150$                    ;GO TO 150$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                150$:
                                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
                                        BR      160$                    ;GO TO 160$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                160$:

                                170$:
                                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                                        BR      180$                    ;GO TO 180$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                180$:

                                ;CHANGE LOOP ADDRESSES
                                        MOV     #190$,$LPADR
                                        MOV     #190$,$LPERR
                                        MOV     RMDC0,360$
                                        MOV     #185$,$ESCAPE   ;;ESCAPE TO 185$ ON ERROR
                                        BR      200$                    ;SKIP TO NEXT OPERATION
                                185$:   MOV     360$,RMDC0              ;RESTORE CYLINDER

                                ;;*****************************************************************
                                ;LOOP #3        READ

                                190$:

                                ;PREPARE DEVICE FOR READ OPERATION
                                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                                        .WORD   054130  ;TASK DESCRIPTOR
                                        BR      200$                    ;GO TO 200$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                200$:
```

```
5633
5634  020254  013737  001432  020616          MOV     RMDC0,360$              ;SAVE CYLINDER
5635  020262  012737  001466  001432          MOV     #822.,RMDC0             ;SEEK TO LAST CYLINDER
5636  020270  012737  000005  001376          MOV     #SEEK!GO,RMCS10         ;WRITE SEEK COMMAND
5637  020276  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5638  020302  000404                          BR      210$                    ;GO TO 210$ IF NO ERROR
5639  020304  000240                          NOP                     ;RETURN HERE IF ERROR
5640  020306  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5641  020310  000137  020614                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5642  020314                          210$:
5643
5644                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5645  020314  004737  042712                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5646  020320  004737  043606                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5647  020324  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5648  020330  000404                          BR      220$                    ;GO TO 220$ IF NO ERROR
5649  020332  000240                          NOP                     ;RETURN HERE IF ERROR
5650  020334  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5651  020336  000137  020614                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5652  020342                          220$:
5653  020342  004737  050700                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
5654  020346  000405                          BR      230$                    ;GO TO 230$ IF NO ERROR
5655  020350  000240                          NOP                     ;RETURN HERE IF ERROR
5656  020352  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5657  020354  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5658  020356  000137  020614                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5659  020362  013737  020616  001432  230$:   MOV     360$,RMDC0              ;RESTORE CYLINDER
5660  020370  012737  000000  001210          MOV     #0,$ESCAPE      ;;ESCAPE TO 0 ON ERROR
5661
5662  020376                          240$:
5663
5664                                  ;READ DATA FROM DEVICE
5665  020376  012737  000071  001376          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
5666  020404  012737  107530  001402          MOV     #BUFTWO+4,RMBA0         ;LOAD STARTING BUFFER ADDRESS
5667  020412  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5668  020416  112722  000006                  MOVB    #RMDA,(R2)+
5669  020422  112722  000032                  MOVB    #RMOF,(R2)+
5670  020426  112722  000034                  MOVB    #RMDC,(R2)+
5671  020432  112722  000004                  MOVB    #RMBA,(R2)+
5672  020436  112722  000002                  MOVB    #RMWC,(R2)+
5673  020442  112722  000000                  MOVB    #RMCS1,(R2)+
5674  020446  112712  000200                  MOVB    #200,(R2)
5675
5676  020452  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5677  020456  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
5678  020460  000240                          NOP                     ;RETURN HERE IF ERROR
5679  020462  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5680  020464  000137  020614                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5681  020470                          250$:
5682
5683
5684                                  ;WAIT FOR READ OPERATION TO COMPLETE
5685  020470  004737  043606                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5686
5687                                  ;GO READ STATUS FOR READ OPERATION
5688  020474  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
```

```
                                        BR      260$                    ;GO TO 260$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                260$:

                                ;CHECK FOR ERRORS DURING READ OPERATION
                                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                                        BR      270$                    ;GO TO 270$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                270$:
                                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
                                        BR      280$                    ;GO TO 280$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                280$:

                                290$:
                                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                                        BR      300$                    ;GO TO 300$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                300$:
                                        JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
                                        .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
                                        .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
                                        BR      310$                    ;GO TO 310$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                310$:

                                350$:
                                        BR      370$
                                360$:   .WORD
                                370$:
                                ;;*****************************************************************
                                ;*TEST 14       WRITE, WRITE CHECK W/ HEAD SWITCHING
                                ;;*****************************************************************
                                TST14:
                                        SCOPE                   ;SCOPE CALL
                                        NOP                     ;START OF TEST
                                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                                        MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                                ;;*****************************************************************
                                ;LOOP #1        FORMAT,WRITE,READ
```

```
5745
5746  020646                                  10$:
5747
5748                                          ;PREPARE THE DEVICE FOR FORMAT OPERATION
5749  020646  004737  037260                          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5750  020652  054130                                  .WORD   054130  ;TASK DESCRIPTOR
5751  020654  000404                                  BR      20$                     ;GO TO 20$ IF NO ERROR
5752  020656  000240                                  NOP                     ;RETURN HERE IF ERROR
5753  020660  104000                                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5754  020662  000137  021640                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5755  020666                                  20$:
5756
5757                                          ;LOAD PARAMETERS AND GENERATE DATA BUFFER
5758  020666  012737  000000  001432                  MOV     #0,RMDC0                ;CYLINDER = 0
5759  020674  012737  000037  001404                  MOV     #037,RMDA0              ;TRACK = 0, SECTOR = 31
5760  020702  012737  106520  001402                  MOV     #BUFONE,RMBA0           ;BUS ADDRESS
5761  020710  012737  176774  001400                  MOV     #<^C<2*<256.+2>>+1>,RMWC0               ;WORD COUNT
5762  020716  012737  010000  001430                  MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5763  020724  012737  000063  001376                  MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
5764
5765                                          ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
5766
5767  020732  004737  040174                          JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
5768  020736  000405                                  BR      25$                     ;GO TO 25$ IF NO ERROR
5769  020740  104401  066704                          TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
5770  020744  104000                                  ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
5771  020746  000137  021640                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5772  020752                                  25$:
5773  020752  012737  070016  001174                  MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
5774  020760  012737  000001  001176                  MOV     #1,$TMP1                ;RANGE OF PATTERN
5775  020766  004737  042100                          JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
5776
5777                                          ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
5778  020772  012702  001533                          MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
5779  020776  112722  000034                          MOVB    #RMDC,(R2)+
5780  021002  112722  000006                          MOVB    #RMDA,(R2)+
5781  021006  112722  000004                          MOVB    #RMBA,(R2)+
5782  021012  112722  000002                          MOVB    #RMWC,(R2)+
5783  021016  112722  000032                          MOVB    #RMOF,(R2)+
5784  021022  112722  000000                          MOVB    #RMCS1,(R2)+
5785  021026  112712  000200                          MOVB    #200,(R2)               ;TERMINATE TABLE
5786  021032                                  30$:
5787
5788                                          ;FORMAT THE DRIVE
5789  021032  004737  043246                          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5790  021036  000404                                  BR      40$                     ;GO TO 40$ IF NO ERROR
5791  021040  000240                                  NOP                     ;RETURN HERE IF ERROR
5792  021042  104000                                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5793  021044  000137  021640                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5794  021050                                  40$:
5795
5796                                          ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
5797  021050  004737  042712                          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5798
5799                                          ;WAIT FOR THE FORMAT TO COMPLETE
5800  021054  004737  043606                          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
```

```
5801
5802                                    ;GO READ STATUS FOR FORMAT OPERATION
5803  021060  004737  042776                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5804  021064  000404                            BR      50$                     ;GO TO 50$ IF NO ERROR
5805  021066  000240                            NOP                     ;RETURN HERE IF ERROR
5806  021070  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5807  021072  000137  021640                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5808  021076                                50$:
5809
5810                                    ;VERIFY NO ERRORS DURING FORMAT
5811  021076  004737  056276                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5812  021102  000405                            BR      60$                     ;GO TO 60$ IF NO ERROR
5813  021104  000240                            NOP                     ;RETURN HERE IF ERROR
5814  021106  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5815  021110  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5816  021112  000137  021640                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5817  021116                                60$:
5818
5819                                    ;MOVE LOOP ADDRESSES TO NEXT OPERATION
5820  021116  012737  021176  001122            MOV     #70$,$LPADR
5821  021124  012737  021176  001124            MOV     #70$,$LPERR
5822                                  ;REGENERATE BUFFER
5823  021132  012737  177000  001400            MOV     #<^C<2*256.>+1>,RMWCO   ;CHANGE WORD COUNT
5824  021140  012737  106520  001402            MOV     #BUFONE,RMBAO           ;CHANGE BUS ADDRESS
5825  021146  012737  070060  001174            MOV     #ZEROS,$TMPO            ;STARTING ADDRESS
5826  021154  012737  000001  001176            MOV     #1,$TMP1                ;RANGE
5827  021162  012737  000060  001376            MOV     #WD,RMCS10              ;WRITE DATA
5828  021170  004737  042100                    JSR     PC,GENBUF               ;GENERATE BUFFER
5829  021174  000410                            BR      80$                     ;SKIP TO WRITE OPERATION
5830
5831                                    ;;*****************************************************************
5832                                    ;LOOP #2        WRITE,READ
5833
5834  021176                                70$:
5835
5836
5837                                    ;PREPARE DEVICE FOR WRITE OPERATION
5838  021176  004737  037260                    JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
5839  021202  054130                            .WORD   054130  ;TASK DESCRIPTOR
5840  021204  000404                            BR      80$                     ;GO TO 80$ IF NO ERROR
5841  021206  000240                            NOP                     ;RETURN HERE IF ERROR
5842  021210  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5843  021212  000137  021640                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5844  021216                                80$:
5845
5846
5847  021216                                120$:
5848
5849                                    ;WRITE DATA TO THE DRIVE
5850  021216  012737  000061  001376            MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
5851  021224  012702  001533                    MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5852  021230  112722  000006                    MOVB    #RMDA,(R2)+
5853  021234  112722  000034                    MOVB    #RMDC,(R2)+
5854  021240  112722  000032                    MOVB    #RMOF,(R2)+
5855  021244  112722  000004                    MOVB    #RMBA,(R2)+
5856  021250  112722  000002                    MOVB    #RMWC,(R2)+
```

```
                                        MOVB    #RMCS1,(R2)+
                                        MOVB    #200,(R2)+              ;TERMINATE TABLE

                                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
                                        BR      130$                    ;GO TO 130$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                130$:

                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
                                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR WRITE COMMAND TO COMPLETE
                                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR WRITE COMMAND
                                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
                                        BR      140$                    ;GO TO 140$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                140$:

                                ;CHECK FOR ERRORS DURING WRITE OPERATION
                                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                                        BR      150$                    ;GO TO 150$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                150$:
                                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
                                        BR      160$                    ;GO TO 160$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                160$:

                                170$:
                                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                                        BR      180$                    ;GO TO 180$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                180$:

                                ;CHANGE LOOP ADDRESSES
                                        MOV     #190$,$LPADR
                                        MOV     #190$,$LPERR
                                        BR      200$                    ;SKIP TO NEXT OPERATION

                                ;;*****************************************************************
                                ;LOOP #3        READ
```

```
5913
5914  021426                          190$:
5915
5916                                  ;PREPARE DEVICE FOR READ OPERATION
5917  021426  004737  037260                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5918  021432  054130                        .WORD   054130  ;TASK DESCRIPTOR
5919  021434  000404                        BR      200$                    ;GO TO 200$ IF NO ERROR
5920  021436  000240                        NOP                     ;RETURN HERE IF ERROR
5921  021440  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5922  021442  000137  021640                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5923  021446                          200$:
5924
5925  021446                          240$:
5926
5927                                  ;READ DATA FROM DEVICE
5928  021446  012737  000051  001376        MOV     #WCD!GO,RMCS10          ;READ DATA COMMAND
5929  021454  012702  001533                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5930  021460  112722  000006                MOVB    #RMDA,(R2)+
5931  021464  112722  000032                MOVB    #RMOF,(R2)+
5932  021470  112722  000034                MOVB    #RMDC,(R2)+
5933  021474  112722  000004                MOVB    #RMBA,(R2)+
5934  021500  112722  000002                MOVB    #RMWC,(R2)+
5935  021504  112722  000000                MOVB    #RMCS1,(R2)+
5936  021510  112712  000200                MOVB    #200,(R2)
5937
5938  021514  004737  043246                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5939  021520  000404                        BR      250$                    ;GO TO 250$ IF NO ERROR
5940  021522  000240                        NOP                     ;RETURN HERE IF ERROR
5941  021524  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5942  021526  000137  021640                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5943  021532                          250$:
5944
5945                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5946  021532  004737  042712                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5947
5948                                  ;WAIT FOR READ OPERATION TO COMPLETE
5949  021536  004737  043606                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5950
5951                                  ;GO READ STATUS FOR READ OPERATION
5952  021542  004737  042776                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5953  021546  000404                        BR      260$                    ;GO TO 260$ IF NO ERROR
5954  021550  000240                        NOP                     ;RETURN HERE IF ERROR
5955  021552  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5956  021554  000137  021640                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5957  021560                          260$:
5958
5959                                  ;CHECK FOR ERRORS DURING READ OPERATION
5960  021560  004737  043772                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5961  021564  000405                        BR      270$                    ;GO TO 270$ IF NO ERROR
5962  021566  000240                        NOP                     ;RETURN HERE IF ERROR
5963  021570  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5964  021572  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5965  021574  000137  021640                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5966  021600                          270$:
5967  021600  004737  056276                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5968  021604  000405                        BR      280$                    ;GO TO 280$ IF NO ERROR
```

```
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                280$:

                290$:
                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                        BR      300$                    ;GO TO 300$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                300$:

                350$:
                ;;*****************************************************************
                ;*TEST 15        WRITE, WRITE CHECK W/ MID-TRANSFER SEEK
                ;;*****************************************************************
                TST15:
                        SCOPE                   ;SCOPE CALL
                        NOP                     ;START OF TEST
                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                        MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                ;;*****************************************************************
                ;LOOP #1         FORMAT,WRITE,READ

                10$:

                ;PREPARE THE DEVICE FOR FORMAT OPERATION
                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                        .WORD   054130  ;TASK DESCRIPTOR
                        BR      20$                     ;GO TO 20$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                20$:

                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
                        MOV     #0,RMDC0                ;CYLINDER = 0
                        MOV     #2037,RMDA0             ;TRACK = 4, SECTOR = 31
                        MOV     #BUF0ME,RMBA0           ;BUS ADDRESS
                        MOV     #^C<2*<256.+2>>+1>,RMWC0                ;WORD COUNT
                        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
                        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND

                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE

                        JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
                        BR      25$                     ;GO TO 25$ IF NO ERROR
                        TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
                        ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
```

```
 6025  021772                          25$:
 6026  021772  012737  070060  001174          MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
 6027  022000  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
 6028  022006  004737  042100                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
 6029
 6030                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
 6031  022012  012702  001533          MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
 6032  022016  112722  000034          MOVB    #RMDC,(R2)+
 6033  022022  112722  000006          MOVB    #RMDA,(R2)+
 6034  022026  112722  000004          MOVB    #RMBA,(R2)+
 6035  022032  112722  000002          MOVB    #RMWC,(R2)+
 6036  022036  112722  000032          MOVB    #RMOF,(R2)+
 6037  022042  112722  000000          MOVB    #RMCS1,(R2)+
 6038  022046  112712  000200          MOVB    #200,(R2)               ;TERMINATE TABLE
 6039  022052                          30$:
 6040
 6041                                  ;FORMAT THE DRIVE
 6042  022052  004737  043246          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
 6043  022056  000404                  BR      40$                     ;GO TO 40$ IF NO ERROR
 6044  022060  000240                  NOP                     ;RETURN HERE IF ERROR
 6045  022062  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
 6046  022064  000137  022660          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 6047  022070                          40$:
 6048
 6049                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
 6050  022070  004737  042712          JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
 6051
 6052                                  ;WAIT FOR THE FORMAT TO COMPLETE
 6053  022074  004737  043606          JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
 6054
 6055                                  ;GO READ STATUS FOR FORMAT OPERATION
 6056  022100  004737  042776          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 6057  022104  000404                  BR      50$                     ;GO TO 50$ IF NO ERROR
 6058  022106  000240                  NOP                     ;RETURN HERE IF ERROR
 6059  022110  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 6060  022112  000137  022660          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 6061  022116                          50$:
 6062
 6063                                  ;VERIFY NO ERRORS DURING FORMAT
 6064  022116  004737  056276          JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
 6065  022122  000405                  BR      60$                     ;GO TO 60$ IF NO ERROR
 6066  022124  000240                  NOP                     ;RETURN HERE IF ERROR
 6067  022126  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 6068  022130  004736                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 6069  022132  000137  022660          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 6070  022136                          60$:
 6071
 6072                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
 6073  022136  012737  022216  001122          MOV     #70$,$LPADR
 6074  022144  012737  022216  001124          MOV     #70$,$LPERR
 6075                                  ;REGENERATE BUFFER
 6076  022152  012737  000060  001376          MOV     #WD,RMCS10              ;WRITE DATA
 6077  022160  012737  177000  001400          MOV     #<^C<2*256.>+1>,RMWC0   ;CHANGE WORD COUNT
 6078  022166  012737  106520  001402          MOV     #BUFONE,RMBA0           ;CHANGE BUS ADDRSS
 6079  022174  012737  070016  001174          MOV     #ONES,$TMP0             ;PATTERN ADDRESS
 6080  022202  012737  000001  001176          MOV     #1,$TMP1                ;PATTERN RANGE
```

```
                                        JSR     PC,GENBUF
                                        BR      80$                     ;SKIP TO WRITE OPERATION

                                ;;**********************************************************************
                                ;LOOP #2        WRITE,READ

                                70$:

                                ;PREPARE DEVICE FOR WRITE OPERATION
                                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                                        .WORD   054130  ;TASK DESCRIPTOR
                                        BR      80$                     ;GO TO 80$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                80$:

                                120$:

                                ;WRITE DATA TO THE DRIVE
                                        MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
                                        MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
                                        MOVB    #RMDA,(R2)+
                                        MOVB    #RMDC,(R2)+
                                        MOVB    #RMOF,(R2)+
                                        MOVB    #RMBA,(R2)+
                                        MOVB    #RMWC,(R2)+
                                        MOVB    #RMCS1,(R2)+
                                        MOVB    #200,(R2)+              ;TERMINATE TABLE

                                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
                                        BR      130$                    ;GO TO 130$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                130$:

                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
                                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR WRITE COMMAND TO COMPLETE
                                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR WRITE COMMAND
                                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
                                        BR      140$                    ;GO TO 140$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                140$:

                                ;CHECK FOR ERRORS DURING WRITE OPERATION
                                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                                        BR      150$                    ;GO TO 150$ IF NO ERROR
```

```
                        NOP                             ;RETURN HERE IF ERROR
                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                150$:
                        JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
                        BR      160$                            ;GO TO 160$ IF NO ERROR
                        NOP                             ;RETURN HERE IF ERROR
                        ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                160$:

                170$:
                        JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
                        BR      180$                            ;GO TO 180$ IF NO ERROR
                        NOP                             ;RETURN HERE IF ERROR
                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                180$:

                ;CHANGE LOOP ADDRESSES
                        MOV     #190$,$LPADR
                        MOV     #190$,$LPERR
                        BR      200$                    ;SKIP TO NEXT OPERATION

                ;;*****************************************************************
                ;LOOP #3         READ

                190$:

                ;PREPARE DEVICE FOR READ OPERATION
                        JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
                        .WORD   054130  ;TASK DESCRIPTOR
                        BR      200$                            ;GO TO 200$ IF NO ERROR
                        NOP                             ;RETURN HERE IF ERROR
                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                200$:

                240$:

                ;READ DATA FROM DEVICE
                        MOV     #WCD!GO,RMCS10          ;READ DATA COMMAND
                        MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
                        MOVB    #RMDA,(R2)+
                        MOVB    #RMOF,(R2)+
                        MOVB    #RMDC,(R2)+
                        MOVB    #RMBA,(R2)+
                        MOVB    #RMWC,(R2)+
                        MOVB    #RMCS1,(R2)+
                        MOVB    #200,(R2)

                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
                        BR      250$                            ;GO TO 250$ IF NO ERROR
```

```
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY PUT SUBROUTINE
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                250$:

                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
                                        JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR READ OPERATION TO COMPLETE
                                        JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR READ OPERATION
                                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
                                        BR      260$                            ;GO TO 260$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY GET SUBROUTINE
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                260$:

                                ;CHECK FOR ERRORS DURING READ OPERATION
                                        JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
                                        BR      270$                            ;GO TO 270$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                270$:
                                        JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
                                        BR      280$                            ;GO TO 280$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                280$:

                                290$:
                                        JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
                                        BR      300$                            ;GO TO 300$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                300$:

                                350$:
                                ;;*****************************************************************
                                ;*TEST 16       WRITE, READ W/ HCE ERROR
                                ;;*****************************************************************
                                TST16:
                                        SCOPE                           ;SCOPE CALL
                                        NOP                             ;START OF TEST
                                        MOV     #STACK,SP               ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
                                        MOV     #16,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
```

```
                                ;;*****************************************************************
                                ;LOOP #1        FORMAT,WRITE,READ

000000                                  MOV     #0,R4                           ;R4=HEADER WORD
                                ;;*****************************************************************
                                ;NOTE:  BSE NOT TESTED
020000                          10$:    MOV     #BIT13,R3                       ;R3=HCE BIT
                                15$:

                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
037260                                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                                        .WORD   054130  ;TASK DESCRIPTOR
000404                                  BR      20$                             ;GO TO 20$ IF NO ERROR
000240                                  NOP                     ;RETURN HERE IF ERROR
104000                                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
000137  024354                         JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                20$:

                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
000000  001432                         MOV     #0,RMDC0                        ;CYLINDER = 0
000000  001404                         MOV     #0,RMDA0                        ;TRACK = 0, SECTOR = 0
106520  001402                         MOV     #BUFONE,RMBA0                   ;BUS ADDRESS
177376  001400                         MOV     #<^C<2+256.>+1>,RMWC0           ;WORD COUNT
010000  001430                         MOV     #FMT16,RMOF0                    ;16 BIT FORMAT
000063  001376                         MOV     #WH!GO,RMCS10                   ;WRITE HEADER AND DATA COMMAND

                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE

040174                                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
000405                                  BR      25$                             ;GO TO 25$ IF NO ERROR
066704                                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
104000                                  ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
024354                                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                25$:
070060  001174                         MOV     #ZEROS,$TMP0                    ;STARTING ADDRESS OF PATTERN
000001  001176                         MOV     #1,$TMP1                        ;RANGE OF PATTERN
042100                                  JSR     PC,GENBUF                       ;GO GENERATE BUFFER FOR FORMAT

                                ;CHANGE HEADER WORD TO FORCE HCE DURING WRITE & READ
106520                                  BIT     R3,BUFONE(R4)                   ;SET OR RESET FOR HCE??
001403                                  BEQ     26$                             ;
106520                                  BIC     R3,BUFONE(R4)                   ;RESET BIT FOR HCE
000402                                  BR      27$
106520                          26$:    BIS     R3,BUFONE(R4)                   ;SET FOR HCE
                                27$:
                                ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
001533                                  MOV     #PUTINX,R2                      ;R2 = BYTE ENTRY POSITION
000034                                  MOVB    #RMDC,(R2)+
000006                                  MOVB    #RMDA,(R2)+
000004                                  MOVB    #RMBA,(R2)+
000002                                  MOVB    #RMWC,(R2)+
000032                                  MOVB    #RMOF,(R2)+
000000                                  MOVB    #RMCS1,(R2)+
000200                                  MOVB    #200,(R2)                       ;TERMINATE TABLE
                                30$:
```

```
                                ;FORMAT THE DRIVE
023122  004737  043246                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
023126  000404                        BR      40$                     ;GO TO 40$ IF NO ERROR
023130  000240                        NOP                     ;RETURN HERE IF ERROR
023132  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
023134  000137  024354                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
023140                          40$:

                                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
023140  004737  042712                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR THE FORMAT TO COMPLETE
023144  004737  043606                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR FORMAT OPERATION
023150  004737  042776                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
023154  000404                        BR      50$                     ;GO TO 50$ IF NO ERROR
023156  000240                        NOP                     ;RETURN HERE IF ERROR
023160  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
023162  000137  024354                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
023166                          50$:

                                ;VERIFY NO ERRORS DURING FORMAT
023166  004737  056276                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
023172  000405                        BR      60$                     ;GO TO 60$ IF NO ERROR
023174  000240                        NOP                     ;RETURN HERE IF ERROR
023176  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
023200  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
023202  000137  024354                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
023206                          60$:

                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
023206  012737  023224  001122            MOV     #70$,$LPADR
023214  012737  023224  001124            MOV     #70$,$LPERR
023222  000410                        BR      80$                     ;SKIP TO WRITE OPERATION

                                ;;*****************************************************************
                                ;LOOP #2        WRITE,READ

023224                          70$:

                                ;PREPARE DEVICE FOR WRITE OPERATION
023224  004737  037260                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
023230  054130                        .WORD   054130  ;TASK DESCRIPTOR
023232  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
023234  000240                        NOP                     ;RETURN HERE IF ERROR
023236  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
023240  000137  024354                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
023244                          80$:

023244                          120$:

                                ;WRITE DATA TO THE DRIVE
023244  012737  106524  001402            MOV     #BUFONE+4,RMBA0         ;CHANGE BUS ADDRESS
```

```
6361  023252  012737  177400  001400          MOV     #<^C256.+1>,RMWCO        ;CHANGE WORD COUNT
6362  023260  012737  000061  001376          MOV     #WD!GO,RMCS1O            ;WRITE DATA COMMAND
6363  023266  012702  001533                  MOV     #PUTINX,R2               ;LOAD PUT REGISTER INDEX TABLE
6364  023272  112722  000006                  MOVB    #RMDA,(R2)+
6365  023276  112722  000034                  MOVB    #RMDC,(R2)+
6366  023302  112722  000032                  MOVB    #RMOF,(R2)+
6367  023306  112722  000004                  MOVB    #RMBA,(R2)+
6368  023312  112722  000002                  MOVB    #RMWC,(R2)+
6369  023316  112722  000000                  MOVB    #RMCS1,(R2)+
6370  023322  112722  000200                  MOVB    #200,(R2)+               ;TERMINATE TABLE
6371
6372  023326  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6373  023332  000404                          BR      130$                     ;GO TO 130$ IF NO ERROR
6374  023334  000240                          NOP                      ;RETURN HERE IF ERROR
6375  023336  104000                          ERROR                    ;ERROR # DEFINED BY PUT SUBROUTINE
6376  023340  000137  024354                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
6377  023344                          130$:
6378
6379                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
6380  023344  004737  042712                  JSR     PC,GETSTS        ;GO TO GETSTS SUBROUTINE
6381
6382                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
6383  023350  004737  043606                  JSR     PC,TIMOUT        ;GO TO TIMOUT SUBROUTINE
6384
6385                                  ;GO READ STATUS FOR WRITE COMMAND
6386  023354  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6387  023360  000404                          BR      140$                     ;GO TO 140$ IF NO ERROR
6388  023362  000240                          NOP                      ;RETURN HERE IF ERROR
6389  023364  104000                          ERROR                    ;ERROR # DEFINED BY GET SUBROUTINE
6390  023366  000137  024354                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
6391  023372                          140$:
6392
6393                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
6394  023372  004737  043772                  JSR     PC,PRIERR        ;GO CHECK FOR PRIMARY ERRORS
6395  023376  000405                          BR      145$                     ;GO TO 145$ IF NO ERROR
6396  023400  000240                          NOP                      ;RETURN HERE IF ERROR
6397  023402  104000                          ERROR                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
6398  023404  004736                          JSR     PC,@(SP)+        ;GO BACK FOR MORE ERROR CHECKS
6399  023406  000137  024354                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
6400  023412                          145$:
6401
6402                                  ;DECODE THE TYPE OF ERROR PRODUCED BY THE BIT POSITION OF R3
6403  023412  005704                          TST     R4                       ;IS THIS THE FIRST HEADER WORD ?
6404  023414  001006                          BNE     146$                     ;NO !!
6405  023416  032703  010000                  BIT     #FMT16,R3                ;IS THIS A FORMAT ERROR ??
6406  023422  001034                          BNE     155$                     ;YES !!
6407  023424  032703  140000                  BIT     #MSE!USE,R3              ;IS THIS A BAD SECTOR ERROR ??
6408  023430  001056                          BNE     165$                     ;YES !!
6409  023432                          146$:
6410
6411                                  ;VERIFY THAT A HEADER COMPARE ERROR WAS DETECTED
6412  023432  032737  000200  001342          BIT     #HCE,RMER1I              ;WAS HCE DETECTED ??
6413  023440  001077                          BNE     175$                     ;YES !!
6414                                  ;HCE WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6415  023442  004737  056276                  JSR     PC,DTASTS        ;GO VERIFY RESULTS OF DATA TRANSFER
6416  023446  000405                          BR      150$                     ;GO TO 150$ IF NO ERROR
```

```
6417  023450  000240                        NOP                     ;RETURN HERE IF ERROR
6418  023452  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6419  023454  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6420  023456  000137  024354                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6421  023462                          150$:
6422  023462  013737  001342  001142        MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
6423  023470  012737  000200  001140        MOV     #HCE,$GDDAT             ;EXPECTED STATUS
6424  023476  010437  001174                MOV     R4,$TMP0        ;R4 = HEADER WORD NUMBER
6425  023502  010337  001176                MOV     R3,$TMP1        ;R3 = BIT POSIITON
6426  023506  104344                        ERROR   344                     ;HCE NOT DETECTED
6427  023510  000137  024354                JMP     350$
6428  023514                          155$:
6429
6430                                  ;VERIFY THAT A FORMAT ERROR WAS DETECTED
6431  023514  032737  000020  001342        BIT     #FER,RMER1I             ;WAS FER DETECTED ??
6432  023522  001046                        BNE     175$                    ;YES !!
6433                                  ;FER WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6434  023524  004737  056276                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6435  023530  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
6436  023532  000240                        NOP                     ;RETURN HERE IF ERROR
6437  023534  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6438  023536  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6439  023540  000137  024354                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6440  023544                          160$:
6441  023544  012737  000020  001140        MOV     #FER,$GDDAT             ;EXPECTED STATUS
6442  023552  013737  001342  001142        MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
6443  023560  104343                        ERROR   343                     ;FER NOT DETECTED
6444  023562  000137  024354                JMP     350$
6445  023566                          165$:
6446
6447                                  ;VERIFY THAT A BAD SECTOR ERROR WAS DETECTED
6448  023566  032737  100000  001370        BIT     #BSE,RMER2I             ;WAS BSE DETECTED ??
6449  023574  001021                        BNE     175$                    ;YES !!
6450                                  ;BSE WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6451  023576  004737  056276                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6452  023602  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
6453  023604  000240                        NOP                     ;RETURN HERE IF ERROR
6454  023606  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6455  023610  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6456  023612  000137  024354                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6457  023616                          170$:
6458  023616  013737  001370  001142        MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
6459  023624  012737  100000  001140        MOV     #BSE,$GDDAT             ;EXPECTED STAIUS
6460  023632  104345                        ERROR   345                     ;BSE NOTDETECTED
6461  023634  000137  024354                JMP     350$
6462  023640                          175$:
6463  023640  004737  044624                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6464  023644  000405                        BR      180$                    ;GO TO 180$ IF NO ERROR
6465  023646  000240                        NOP                     ;RETURN HERE IF ERROR
6466  023650  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6467  023652  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6468  023654  000137  024354                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6469  023660                          180$:
6470
6471                                  ;CHANGE LOOP ADDRESSES
6472  023660  012737  023674  001122        MOV     #190$,$LPADR
```

```
6473  023666  012737  023674  001124          MOV     #190$,$LPERR
6474
6475                                  ;;*****************************************************************
6476                                  ;LOOP #3        READ
6477
6478  023674                          190$:
6479
6480                                  ;PREPARE DEVICE FOR READ OPERATION
6481  023674  004737  037260                 JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6482  023700  054130                         .WORD   054130  ;TASK DESCRIPTOR
6483  023702  000404                         BR      200$                    ;GO TO 200$ IF NO ERROR
6484  023704  000240                         NOP                     ;RETURN HERE IF ERROR
6485  023706  104000                         ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6486  023710  000137  024354                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6487  023714                          200$:
6488
6489  023714                          240$:
6490
6491                                  ;READ DATA FROM DEVICE
6492  023714  012737  107530  001402        MOV     #BUFTWO+4,RMBAO         ;CHANGE BUS ADDRESS
6493  023722  012737  177400  001400        MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
6494  023730  012737  000071  001376        MOV     #RD!GO,RMCS1O           ;READ DATA COMMAND
6495  023736  012702  001533                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
6496  023742  112722  000006                MOVB    #RMDA,(R2)+
6497  023746  112722  000032                MOVB    #RMOF,(R2)+
6498  023752  112722  000034                MOVB    #RMDC,(R2)+
6499  023756  112722  000004                MOVB    #RMBA,(R2)+
6500  023762  112722  000002                MOVB    #RMWC,(R2)+
6501  023766  112722  000000                MOVB    #RMCS1,(R2)+
6502  023772  112712  000200                MOVB    #200,(R2)
6503
6504  023776  004737  043246                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6505  024002  000404                        BR      250$                    ;GO TO 250$ IF NO ERROR
6506  024004  000240                        NOP                     ;RETURN HERE IF ERROR
6507  024006  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6508  024010  000137  024354                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6509  024014                          250$:
6510
6511                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
6512  024014  004737  042712                 JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6513
6514                                  ;WAIT FOR READ OPERATION TO COMPLETE
6515  024020  004737  043606                 JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
6516
6517                                  ;GO READ STATUS FOR READ OPERATION
6518  024024  004737  042776                 JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6519  024030  000404                         BR      260$                    ;GO TO 260$ IF NO ERROR
6520  024032  000240                         NOP                     ;RETURN HERE IF ERROR
6521  024034  104000                         ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6522  024036  000137  024354                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6523  024042                          260$:
6524
6525                                  ;CHECK FOR ERRORS DURING READ OPERATION
6526  024042  004737  043772                 JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6527  024046  000405                         BR      265$                    ;GO TO 265$ IF NO ERROR
6528  024050  000240                         NOP                     ;RETURN HERE IF ERROR
```

```
6529  024052  104000                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
6530  024054  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6531  024056  000137  024354                JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
6532  024062                          265$:
6533                                  ;DECODE THE TYPE OF ERROR PRODUCED BY THE BIT POSITION OF R3
6534  024062  005704                        TST     R4                              ;IS THIS THE FIRST HEADER WORD ?
6535  024064  001006                        BNE     266$                            ;NO !!
6536  024066  032703  010000                BIT     #FMT16,R3                       ;IS THIS A FORMAT ERROR ??
6537  024072  001034                        BNE     275$                            ;YES !!
6538  024074  032703  140000                BIT     #MSE!USE,R3                     ;IS THIS A BAD SECTOR ERROR ??
6539  024100  001056                        BNE     285$                            ;YES !!
6540  024102                          266$:
6541
6542                                  ;VERIFY THAT A HEADER COMPARE ERROR WAS DETECTED
6543  024102  032737  000200  001342        BIT     #HCE,RMER1I                     ;WAS HCE DETECTED ??
6544  024110  001077                        BNE     295$                            ;YES !!
6545                                  ;HCE WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6546  024112  004737  056276                JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6547  024116  000405                        BR      270$                            ;GO TO 270$ IF NO ERROR
6548  024120  000240                        NOP                             ;RETURN HERE IF ERROR
6549  024122  104000                        ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
6550  024124  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6551  024126  000137  024354                JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
6552  024132                          270$:
6553  024132  013737  001342  001142        MOV     RMER1I,$BDDAT                   ;RECEIVED STATUS
6554  024140  012737  000200  001140        MOV     #HCE,$GDDAT                     ;EXPECTED STATUS
6555  024146  010437  001174                MOV     R4,$TMP0                ;R4 = HEADER WORD NUMBER
6556  024152  010337  001176                MOV     R3,$TMP1                ;R3 = BIT POSITION
6557  024156  104344                        ERROR   344                             ;HCE NOT DETECTED
6558  024160  000137  024354                JMP     350$
6559  024164                          275$:
6560
6561                                  ;VERIFY THAT A FORMAT ERROR WAS DETECTED
6562  024164  032737  000020  001342        BIT     #FER,RMER1I                     ;WAS FER DETECTED ??
6563  024172  001046                        BNE     295$                            ;YES !!
6564                                  ;FER WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6565  024174  004737  056276                JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6566  024200  000405                        BR      280$                            ;GO TO 280$ IF NO ERROR
6567  024202  000240                        NOP                             ;RETURN HERE IF ERROR
6568  024204  104000                        ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
6569  024206  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6570  024210  000137  024354                JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
6571  024214                          280$:
6572  024214  012737  000020  001140        MOV     #FER,$GDDAT                     ;EXPECTED STATUS
6573  024222  013737  001342  001142        MOV     RMER1I,$BDDAT                   ;RECEIVED STATUS
6574  024230  104343                        ERROR   343                             ;FER NOT DETECTED
6575  024232  000137  024354                JMP     350$
6576  024236                          285$:
6577
6578                                  ;VERIFY THAT A BAD SECTOR ERROR WAS DETECTED
6579  024236  032737  100000  001370        BIT     #BSE,RMER2I                     ;WAS BSE DETECTED ??
6580  024244  001021                        BNE     295$                            ;YES !!
6581                                  ;BSE WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6582  024246  004737  056276                JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6583  024252  000405                        BR      290$                            ;GO TO 290$ IF NO ERROR
6584  024254  000240                        NOP                             ;RETURN HERE IF ERROR
```

```
                                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                290$:
                                        MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
                                        MOV     #BSE,$GDDAT             ;EXPECTED STAIUS
                                        ERROR   345                     ;BSE NOTDETECTED
                                        JMP     350
                                295$:
                                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                                        BR      300$                    ;GO TO 300$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                300$:
                                        ASR     R3                      ;SHIFT TO NEXT BIT
                                        BNE     310$                    ;REPEAT IF NOT DONE
                                        TST     R4                      ;SECOND HEADER WORD DONE??
                                        BNE     350$                    ;YES!!
                                        MOV     #2,R4                   ;SETUP FOR SECOND HEADER WORD
                                ;;**************************************************************
                                ;NOTE:  BSE NOT TESTED
                                        MOV     #BIT13,R3
                                310$:   JMP     15$
                                350$:
                                        MOV     #10$,$LPADR             ;CHANGE LOOP TO START OF TEST
                                        MOV     #10$,$LPERR
                                ;;**************************************************************
                                ;*TEST 17        WRITE, READ W/ HCI
                                ;;**************************************************************
                                TST17:
                                        SCOPE                   ;SCOPE CALL
                                        NOP                     ;START OF TEST
                                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                                        MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                                ;;**************************************************************
                                ;LOOP #1        FORMAT,WRITE,READ

                                10$:    MOV     #0,R4                   ;HEADER WORD
                                        MOV     #1,R3                   ;HCE BIT

                                15$:
                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
                                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                                        .WORD   054130  ;TASK DESCRIPTOR
                                        BR      20$                     ;GO TO 20$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                20$:
```

```
6641                                    ;LOAD PARAMETERS AND GENERATE DATA BUFFER
6642  024446  012737  000000  001432            MOV     #0,RMDC0                ;CYLINDER = 0
6643  024454  012737  000000  001404            MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
6644  024462  012737  106520  001402            MOV     #BUFONE,RMBA0           ;BUS ADDRESS
6645  024470  012737  177376  001400            MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
6646  024476  012737  010000  001430            MOV     #FMT16,RMOF0            ;16 BIT FORMAT
6647  024504  012737  000063  001376            MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
6648
6649                                    ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
6650
6651  024512  004737  040174                    JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
6652  024516  000405                            BR      25$                     ;GO TO 25$ IF NO ERROR
6653  024520  104401  066704                    TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
6654  024524  104000                            ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
6655  024526  000137  025640                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6656  024532                                25$:
6657  024532  012737  070016  001174            MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
6658  024540  012737  000001  001176            MOV     #1,$TMP1                ;RANGE OF PATTERN
6659  024546  004737  042100                    JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
6660
6661                                    ;CHANGE HEADER WORD TO FORCE ERROR DURING WRITE
6662  024552  030364  106520                    BIT     R3,BUFONE(R4)           ;SET OR RESET BIT??
6663  024556  001403                            BEQ     27$
6664  024560  040364  106520                    BIC     R3,BUFONE(R4)           ;RESET BIT
6665  024564  000402                            BR      28$
6666  024566  050364  106520           27$:     BIS     R3,BUFONE(R4)           ;SET BIT
6667  024572                                28$:
6668                                    ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
6669  024572  012702  001533                    MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
6670  024576  112722  000034                    MOVB    #RMDC,(R2)+
6671  024602  112722  000006                    MOVB    #RMDA,(R2)+
6672  024606  112722  000004                    MOVB    #RMBA,(R2)+
6673  024612  112722  000002                    MOVB    #RMWC,(R2)+
6674  024616  112722  000032                    MOVB    #RMOF,(R2)+
6675  024622  112722  000000                    MOVB    #RMCS1,(R2)+
6676  024626  112712  000200                    MOVB    #200,(R2)               ;TERMINATE TABLE
6677  024632                                30$:
6678
6679                                    ;FORMAT THE DRIVE
6680  024632  004737  043246                    JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6681  024636  000404                            BR      40$                     ;GO TO 40$ IF NO ERROR
6682  024640  000240                            NOP                     ;RETURN HERE IF ERROR
6683  024642  104000                            ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6684  024644  000137  025640                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6685  024650                                40$:
6686
6687                                    ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
6688  024650  004737  042712                    JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
6689
6690                                    ;WAIT FOR THE FORMAT TO COMPLETE
6691  024654  004737  043606                    JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
6692
6693                                    ;GO READ STATUS FOR FORMAT OPERATION
6694  024660  004737  042776                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6695  024664  000404                            BR      50$                     ;GO TO 50$ IF NO ERROR
6696  024666  000240                            NOP                     ;RETURN HERE IF ERROR
```

```
6697  024670  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6698  024672  000137  025640                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6699  024676                          50$:
6700
6701                                  ;VERIFY NO ERRORS DURING FORMAT
6702  024676  004737  056276                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6703  024702  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
6704  024704  000240                          NOP                     ;RETURN HERE IF ERROR
6705  024706  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6706  024710  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6707  024712  000137  025640                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6708  024716                          60$:
6709
6710                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
6711  024716  012737  024734  001122          MOV     #70$,$LPADR
6712  024724  012737  024734  001124          MOV     #70$,$LPERR
6713  024732  000410                          BR      80$                     ;SKIP TO WRITE OPERATION
6714
6715                                  ;;*****************************************************************
6716                                  ;LOOP #2        WRITE,READ
6717
6718  024734                          70$:
6719
6720
6721                                  ;PREPARE DEVICE FOR WRITE OPERATION
6722  024734  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6723  024740  054130                          .WORD   054130  ;TASK DESCRIPTOR
6724  024742  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
6725  024744  000240                          NOP                     ;RETURN HERE IF ERROR
6726  024746  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6727  024750  000137  025640                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6728  024754                          80$:
6729
6730
6731  024754                          120$:
6732
6733                                  ;WRITE DATA TO THE DRIVE
6734  024754  012737  012000  001430          MOV     #HCI!FMT16,RMOF0        ;INHIBIT HEADER COMPARE
6735  024762  012737  000061  001376          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
6736  024770  012737  106524  001402          MOV     #BUFONE+4,RMBA0         ;LOAD STARTING BUFFER ADDRESS
6737  024776  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
6738  025002  112722  000006                  MOVB    #RMDA,(R2)+
6739  025006  112722  000034                  MOVB    #RMDC,(R2)+
6740  025012  112722  000032                  MOVB    #RMOF,(R2)+
6741  025016  112722  000004                  MOVB    #RMBA,(R2)+
6742  025022  112722  000002                  MOVB    #RMWC,(R2)+
6743  025026  112722  000000                  MOVB    #RMCS1,(R2)+
6744  025032  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
6745
6746  025036  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6747  025042  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
6748  025044  000240                          NOP                     ;RETURN HERE IF ERROR
6749  025046  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6750  025050  000137  025640                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6751  025054                          130$:
6752
```

```
                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
025054  004737  042712                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR WRITE COMMAND TO COMPLETE
025060  004737  043606                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR WRITE COMMAND
025064  004737  042776                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
025070  000404                        BR      140$                    ;GO TO 140$ IF NO ERROR
025072  000240                        NOP                     ;RETURN HERE IF ERROR
025074  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
025076  000137  025640                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
025102                          140$:

                                ;CHECK FOR ERRORS DURING WRITE OPERATION
025102  004737  043772                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
025106  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
025110  000240                        NOP                     ;RETURN HERE IF ERROR
025112  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
025114  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
025116  000137  025640                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
025122                          150$:
025122  032737  000220  001342        BIT     #HCE!FER,RMER1I         ;ANY ERROR??
025130  001407                        BEQ     151$
025132  013737  001342  001142        MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
025140  042737  177557  001142        BIC     #^C<HCE!FER>,$BDDAT     ;CLEAR DONT CARES
025146  000412                        BR      152$
025150  032737  100000  001370  151$: BIT     #BSE,RMER2I             ;ANY BAD SECTOR ERRROR ??
025156  001414                        BEQ     155$                    ;NO !!
025160  013737  001370  001142        MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
025166  042737  077777  001142        BIC     #^CBSE,$BDDAT           ;CLEAR DONT CARES
025174  012737  000000  001140  152$: MOV     #0,$GDDAT               ;EXPECTED STATUS
025202  104346                        ERROR   346                     ;HCE W/HCI SET
025204  000137  025640                JMP     350$
025210                          155$:
025210  004737  056276                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
025214  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
025216  000240                        NOP                     ;RETURN HERE IF ERROR
025220  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
025222  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
025224  000137  025640                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
025230                          160$:

025230                          170$:
025230  004737  044624                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
025234  000405                        BR      180$                    ;GO TO 180$ IF NO ERROR
025236  000240                        NOP                     ;RETURN HERE IF ERROR
025240  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
025242  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
025244  000137  025640                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
025250                          180$:

                                ;CHANGE LOOP ADDRESSES
025250  012737  025264  001122        MOV     #190$,$LPADR
025256  012737  025264  001124        MOV     #190$,$LPERR
```

```
                                        ;;*****************************************************************
                                        ;LOOP #3        READ

025264                                  190$:

                                        ;PREPARE DEVICE FOR READ OPERATION
025264  004737  037260                          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
025270  054130                                  .WORD   054130  ;TASK DESCRIPTOR
025272  000404                                  BR      200$                    ;GO TO 200$ IF NO ERROR
025274  000240                                  NOP                     ;RETURN HERE IF ERROR
025276  104000                                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
025300  000137  025640                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
025304                                  200$:

025304                                  240$:

                                        ;READ DATA FROM DEVICE
025304  012737  000071  001376                  MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
025312  012737  107530  001402                  MOV     #BUFTWO+4,RMBAO         ;LOAD STARTING BUFFER ADDRESS
025320  012702  001533                          MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
025324  112722  000006                          MOVB    #RMDA,(R2)+
025330  112722  000032                          MOVB    #RMOF,(R2)+
025334  112722  000034                          MOVB    #RMDC,(R2)+
025340  112722  000004                          MOVB    #RMBA,(R2)+
025344  112722  000002                          MOVB    #RMWC,(R2)+
025350  112722  000000                          MOVB    #RMCS1,(R2)+
025354  112712  000200                          MOVB    #200,(R2)

025360  004737  043246                          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
025364  000404                                  BR      250$                    ;GO TO 250$ IF NO ERROR
025366  000240                                  NOP                     ;RETURN HERE IF ERROR
025370  104000                                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
025372  000137  025640                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
025376                                  250$:

                                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
025376  004737  042712                          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                        ;WAIT FOR READ OPERATION TO COMPLETE
025402  004737  043606                          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                        ;GO READ STATUS FOR READ OPERATION
025406  004737  042776                          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
025412  000404                                  BR      260$                    ;GO TO 260$ IF NO ERROR
025414  000240                                  NOP                     ;RETURN HERE IF ERROR
025416  104000                                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
025420  000137  025640                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
025424                                  260$:

                                        ;CHECK FOR ERRORS DURING READ OPERATION
025424  004737  043772                          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
025430  000405                                  BR      270$                    ;GO TO 270$ IF NO ERROR
025432  000240                                  NOP                     ;RETURN HERE IF ERROR
025434  104000                                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
025436  004736                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
025440  000137  025640                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
```

```
6865  025444
6866  025444  032737  000220  001342          270$:   BIT     #HCE!FER,RMER1I         ;ANY ERROR??
6867  025452  001407                                  BEQ     271$                    ;NO!!
6868  025454  013737  001342  001142                  MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
6869  025462  042737  177557  001142                  BIC     #↑C<HCE!FER>,$BDDAT     ;CLEAR DONT CARES
6870  025470  000412                                  BR      272$
6871  025472  032737  100000  001370          271$:   BIT     #BSE,RMER2I             ;ANY BAD SECTOR ERROR ??
6872  025500  001414                                  BEQ     275$                    ;NO !!
6873  025502  013737  001370  001142                  MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
6874  025510  042737  077777  001142                  BIC     #↑CBSE,$BDDAT           ;CLEAR DONT CARES
6875  025516  012737  000000  001140          272$:   MOV     #0,$GDDAT               ;EXPECTED STATUS
6876  025524  104346                                  ERROR   346                     ;HCE W/HCI SET
6877  025526  000137  025640                          JMP     350$
6878  025532                                  275$:
6879  025532  004737  056276                          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6880  025536  000405                                  BR      280$                    ;GO TO 280$ IF NO ERROR
6881  025540  000240                                  NOP                     ;RETURN HERE IF ERROR
6882  025542  104000                                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6883  025544  004736                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6884  025546  000137  025640                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6885  025552                                  280$:
6886
6887  025552                                  290$:
6888  025552  004737  044624                          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6889  025556  000405                                  BR      300$                    ;GO TO 300$ IF NO ERROR
6890  025560  000240                                  NOP                     ;RETURN HERE IF ERROR
6891  025562  104000                                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6892  025564  004736                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6893  025566  000137  025640                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6894  025572                                  300$:
6895  025572  004737  042344                          JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
6896  025576  106524                                  .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
6897  025600  107530                                  .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
6898  025602  000404                                  BR      310$                    ;GO TO 310$ IF NO ERROR
6899  025604  000240                                  NOP                     ;RETURN HERE IF ERROR
6900  025606  104000                                  ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
6901  025610  000137  025640                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6902  025614                                  310$:
6903
6904  025614  006303                                  ASL     R3                      ;SHIFT HCE BIT
6905  025616  001006                                  BNE     320$                    ;CONTINUE IF NOT DONE
6906  025620  005704                                  TST     R4                      ;SECOND HEADER DONE??
6907  025622  001006                                  BNE     350$                    ;YES!!
6908  025624  012703  000001                          MOV     #1,R3                   ;START WITH BIT 0
6909  025630  012704  000002                          MOV     #2,R4                   ;DO SECOND HEADER WORD
6910  025634                                  320$:
6911  025634  000137  024426                          JMP     15$
6912  025640                                  350$:
6913  025640  012737  024416  001122                  MOV     #10$,$LPADR             ;CHANGE LOOP TO START OF TEST
6914  025646  012737  024416  001124                  MOV     #10$,$LPERR
6915                                  ;;*****************************************************************
6916                                  ;*TEST 20        WRITE, READ W/ IVC ERROR
6917                                  ;;*****************************************************************
6918  025654                                  TST20:
6919  025654  000004                                  SCOPE                   ;SCOPE CALL
6920  025656  000240                                  NOP                     ;START OF TEST
```

```
                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                        MOV     #20,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
                        MOV     #0,RMDC0                ;CYLINDER = *
                        MOV     #0,RMDA0                ;TRACK = *, SECTOR = *
                        MOV     #BUFONE,RMBA0           ;BUS ADDRESS
                        MOV     #<^C256+1>,RMWC0        ;WORD COUNT
                        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
                        MOV     #WD,RMCS10              ;WRITE HEADER AND DATA COMMAND

                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
                        MOV     #70$,$LPADR
                        MOV     #70$,$LPERR
                        BR      80$                     ;SKIP TO WRITE OPERATION

                ;;*****************************************************************
                ;LOOP #2        WRITE,READ

                70$:

                ;PREPARE DEVICE FOR WRITE OPERATION
                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                        .WORD   054130  ;TASK DESCRIPTOR
                        BR      80$                     ;GO TO 80$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                80$:

                ;RESET VOLUME VALID
                        MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
                        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
                        MOV     #DMD,RMMR10     ;RMMR1=DMD
                        JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
                        BR      90$             ;GO TO 90$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
                        JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
                90$:
                        MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
                        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
                        MOV     #0,RMMR10       ;RMMR1=0
                        JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
                        BR      100$            ;GO TO 100$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
                        JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
                100$:
```

```
                                        120$:

                                        ;WRITE DATA TO THE DRIVE
026110  012737  000061  001376                MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
026116  012737  106524  001402                MOV     #BUFONE+4,RMBA0         ;LOAD STARTING BUFFER ADDRESS
026124  012702  001533                        MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
026130  112722  000006                        MOVB    #RMDA,(R2)+
026134  112722  000034                        MOVB    #RMDC,(R2)+
026140  112722  000032                        MOVB    #RMOF,(R2)+
026144  112722  000004                        MOVB    #RMBA,(R2)+
026150  112722  000002                        MOVB    #RMWC,(R2)+
026154  112722  000000                        MOVB    #RMCS1,(R2)+
026160  112722  000200                        MOVB    #200,(R2)+              ;TERMINATE TABLE

026164  004737  043246                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
026170  000404                              BR      130$                    ;GO TO 130$ IF NO ERROR
026172  000240                              NOP                     ;RETURN HERE IF ERROR
026174  104000                              ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
026176  000137  026746                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
026202                                  130$:

                                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
026202  004737  042712                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                        ;WAIT FOR WRITE COMMAND TO COMPLETE
026206  004737  043606                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                        ;GO READ STATUS FOR WRITE COMMAND
026212  004737  042776                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
026216  000404                              BR      140$                    ;GO TO 140$ IF NO ERROR
026220  000240                              NOP                     ;RETURN HERE IF ERROR
026222  104000                              ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
026224  000137  026746                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
026230                                  140$:

                                        ;CHECK FOR ERRORS DURING WRITE OPERATION
026230  004737  043772                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
026234  000405                              BR      150$                    ;GO TO 150$ IF NO ERROR
026236  000240                              NOP                     ;RETURN HERE IF ERROR
026240  104000                              ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
026242  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
026244  000137  026746                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
026250                                  150$:
026250  032737  010000  001370                BIT     #IVC,RMER2I             ;WAS "IVC" DETECTED??
026256  001024                              BNE     170$                    ;YES!!
026260  004737  056276                      JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
026264  000405                              BR      160$                    ;GO TO 160$ IF NO ERROR
026266  000240                              NOP                     ;RETURN HERE IF ERROR
026270  104000                              ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
026272  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
026274  000137  026746                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
026300                                  160$:

026300  013737  001370  001142                MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
026306  013737  001370  001140                MOV     RMER2I,$GDDAT           ;EXPECTED STATUS
```

```
 7033  026314  052737  010000  001140          BIS     #IVC,$GDDAT
 7034  026322  104342                          ERROR   342                     ;IVC NOT DETECTED
 7035  026324  000137  026746                  JMP     350$
 7036  026330                          170$:
 7037  026330  004737  044624                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 7038  026334  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
 7039  026336  000240                          NOP                     ;RETURN HERE IF ERROR
 7040  026340  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 7041  026342  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7042  026344  000137  026746                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 7043  026350                          180$:
 7044
 7045                                  ;CHANGE LOOP ADDRESSES
 7046  026350  012737  026366  001122          MOV     #190$,$LPADR
 7047  026356  012737  026366  001124          MOV     #190$,$LPERR
 7048  026364  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
 7049
 7050                                  ;;*****************************************************************
 7051                                  ;LOOP #3        READ
 7052
 7053  026366                          190$:
 7054
 7055                                  ;PREPARE DEVICE FOR READ OPERATION
 7056  026366  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 7057  026372  054130                          .WORD   054130  ;TASK DESCRIPTOR
 7058  026374  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
 7059  026376  000240                          NOP                     ;RETURN HERE IF ERROR
 7060  026400  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 7061  026402  000137  026746                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 7062  026406                          200$:
 7063
 7064                                  ;RESET VOLUME VALID
 7065  026406  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
 7066  026414  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
 7067  026422  012737  000001  001422          MOV     #DMD,RMMR10     ;RMMR1=DMD
 7068  026430  004737  043246                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
 7069  026434  000404                          BR      210$            ;GO TO 210$ IF NO ERROR
 7070  026436  000240                          NOP                     ;RETURN HERE IF ERROR
 7071  026440  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
 7072  026442  000137  026746                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
 7073  026446                          210$:
 7074  026446  112737  000024  001533          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
 7075  026454  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
 7076  026462  012737  000000  001422          MOV     #0,RMMR10       ;RMMR1=0
 7077  026470  004737  043246                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
 7078  026474  000404                          BR      220$            ;GO TO 220$ IF NO ERROR
 7079  026476  000240                          NOP                     ;RETURN HERE IF ERROR
 7080  026500  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
 7081  026502  000137  026746                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
 7082  026506                          220$:
 7083  026506                          240$:
 7084
 7085                                  ;READ DATA FROM DEVICE
 7086  026506  012737  000071  001376          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
 7087  026514  012737  107530  001402          MOV     #BUFTWO+4,RMBA0         ;LOAD STARTING BUFFER ADDRESS
 7088  026522  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
```

```
7089  026526  112722  000006                  MOVB    #RMDA,(R2)+
7090  026532  112722  000032                  MOVB    #RMOF,(R2)+
7091  026536  112722  000034                  MOVB    #RMDC,(R2)+
7092  026542  112722  000004                  MOVB    #RMBA,(R2)+
7093  026546  112722  000002                  MOVB    #RMWC,(R2)+
7094  026552  112722  000000                  MOVB    #RMCS1,(R2)+
7095  026556  112712  000200                  MOVB    #200,(R2)
7096
7097  026562  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7098  026566  000404                          BR      250$            ;GO TO 250$ IF NO ERROR
7099  026570  000240                          NOP                     ;RETURN HERE IF ERROR
7100  026572  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7101  026574  000137  026746                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7102  026600                          250$:
7103
7104                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7105  026600  004737  042712                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7106
7107                                  ;WAIT FOR READ OPERATION TO COMPLETE
7108  026604  004737  043606                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7109
7110                                  ;GO READ STATUS FOR READ OPERATION
7111  026610  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7112  026614  000404                          BR      260$            ;GO TO 260$ IF NO ERROR
7113  026616  000240                          NOP                     ;RETURN HERE IF ERROR
7114  026620  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7115  026622  000137  026746                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7116  026626                          260$:
7117
7118                                  ;CHECK FOR ERRORS DURING READ OPERATION
7119  026626  004737  043772                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7120  026632  000405                          BR      270$            ;GO TO 270$ IF NO ERROR
7121  026634  000240                          NOP                     ;RETURN HERE IF ERROR
7122  026636  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7123  026640  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7124  026642  000137  026746                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7125  026646                          270$:
7126  026646  032737  010000  001370          BIT     #IVC,RMER2I             ;WAS "IVC" DETECTED??
7127  026654  001024                          BNE     290$            ;YES!!
7128  026656  004737  056276                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7129  026662  000405                          BR      280$            ;GO TO 280$ IF NO ERROR
7130  026664  000240                          NOP                     ;RETURN HERE IF ERROR
7131  026666  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7132  026670  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7133  026672  000137  026746                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7134  026676                          280$:
7135  026676  013737  001370  001140          MOV     RMER2I,$GDDAT           ;EXPECTED STATUS
7136  026704  052737  010000  001140          BIS     #IVC,$GDDAT
7137  026712  013737  001370  001142          MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
7138  026720  104342                          ERROR   342             ;IVC NOT DETECTED
7139  026722  000137  026746                  JMP     350$
7140
7141  026726                          290$:
7142  026726  004737  044624                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7143  026732  000405                          BR      300$            ;GO TO 300$ IF NO ERROR
7144  026734  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
                                        ;ERROR # DEFINED BY SECERR SUBROUTINE
                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                300$:

                350$:
                        MOV     #70$,$LPADR             ;CHANGE LOOP TO START OF TEST
                        MOV     #70$,$LPERR
                ;;*****************************************************************
                ;*TEST 21        WRITE, READ W/ ABORT
                ;;*****************************************************************
                TST21:
                        SCOPE                   ;SCOPE CALL
                        NOP                     ;START OF TEST
                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                        MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
                        MOV     #0,RMDC0                ;CYLINDER = 0
                        MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
                        MOV     #BUFONE,RMBA0           ;BUS ADDRESS
                        MOV     #<-C256.+1>,RMWC0       ;WORD COUNT
                        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
                        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND

                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
                        MOV     #70$,$LPADR
                        MOV     #70$,$LPERR
                        BR      80$                     ;SKIP TO WRITE OPERATION

                ;;*****************************************************************
                ;LOOP #2        WRITE,READ

                70$:

                ;PREPARE DEVICE FOR WRITE OPERATION
                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                        .WORD   054130  ;TASK DESCRIPTOR
                        BR      80$                     ;GO TO 80$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                80$:

                ;SET UNSAFE ERROR
                        MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
                        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
                        MOV     #UNS,RMER10     ;RMER1 OUTPUT BUFFER = UNS
                        JSR     PC,PUT          ;WRITE RMER1 VIA PUT SUB
```

```
7201  027144  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
7202  027146  000240                          NOP                     ;RETURN HERE TO REPORT ERROR
7203  027150  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
7204  027152  000137  027730                  JMP     350$    ;GO TO 350$ IF ERROR WAS FOUND
7205  027156                          90$:
7206
7207  027156                          120$:
7208
7209                                  ;WRITE DATA TO THE DRIVE
7210  027156  012737  000061  001376          MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
7211  027164  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7212  027170  112722  000006                  MOVB    #RMDA,(R2)+
7213  027174  112722  000034                  MOVB    #RMDC,(R2)+
7214  027200  112722  000032                  MOVB    #RMOF,(R2)+
7215  027204  112722  000004                  MOVB    #RMBA,(R2)+
7216  027210  112722  000002                  MOVB    #RMWC,(R2)+
7217  027214  112722  000000                  MOVB    #RMCS1,(R2)+
7218  027220  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
7219
7220
7221                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7222  027224  004737  042712                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7223  027230  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7224  027234  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
7225  027236  000240                          NOP                     ;RETURN HERE IF ERROR
7226  027240  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7227  027242  000137  027730                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7228  027246                          130$:
7229  027246  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7230  027252  000404                          BR      135$                    ;GO TO 135$ IF NO ERROR
7231  027254  000240                          NOP                     ;RETURN HERE IF ERROR
7232  027256  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7233  027260  000137  027730                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7234  027264  032737  020000  001340  135$:   BIT     #PIP,RMDSI              ;WAS COMMAND EXECUTED?
7235  027272  001412                          BEQ     136$
7236  027274  013737  001340  001140          MOV     RMDSI,$GDDAT            ;EXPECTED STATUS
7237  027302  042737  020000  001140          BIC     #PIP,$GDDAT
7238  027310  013737  001340  001142          MOV     RMDSI,$BDDAT            ;RECEIVED STATUS
7239  027316  104347                          ERROR   347                     ;NO ABORT
7240  027320                          136$:
7241
7242                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
7243  027320  004737  043606                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7244
7245                                  ;GO READ STATUS FOR WRITE COMMAND
7246  027324  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7247  027330  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
7248  027332  000240                          NOP                     ;RETURN HERE IF ERROR
7249  027334  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7250  027336  000137  027730                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7251  027342                          140$:
7252
7253                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
7254  027342  004737  043772                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7255  027346  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
7256  027350  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
027352  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
027354  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
027356  000137  027730                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
027362                          150$:

027362  004737  044624                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
027366  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
027370  000240                          NOP                     ;RETURN HERE IF ERROR
027372  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
027374  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
027376  000137  027730                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
027402                          180$:

                                ;CHANGE LOOP ADDRESSES
027402  012737  027420  001122          MOV     #190$,$LPADR
027410  012737  027420  001124          MOV     #190$,$LPERR
027416  000410                          BR      200$                    ;SKIP TO NEXT OPERATION

                                ;;*****************************************************************
                                ;LOOP #3        READ

027420                          190$:

                                ;PREPARE DEVICE FOR READ OPERATION
027420  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
027424  054130                          .WORD   054130  ;TASK DESCRIPTOR
027426  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
027430  000240                          NOP                     ;RETURN HERE IF ERROR
027432  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
027434  000137  027730                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
027440                          200$:
                                ;SET UNSAFE ERROR
027440  112737  000014  001533          MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
027446  112737  000200  001534          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
027454  012737  040000  001412          MOV     #UNS,RMER10     ;RMER1 OUTPUT BUFFER = UNS
027462  004737  043246                  JSR     PC,PUT          ;WRITE RMER1 VIA PUT SUB
027466  000404                          BR      210$    ;GO TO 210$ IF NO ERROR
027470  000240                          NOP                     ;RETURN HERE TO REPORT ERROR
027472  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
027474  000137  027730                  JMP     350$    ;GO TO 350$ IF ERROR WAS FOUND
027500                          210$:

027500                          240$:

                                ;READ DATA FROM DEVICE
027500  012737  000071  001376          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
027506  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
027512  112722  000006                  MOVB    #RMDA,(R2)+
027516  112722  000032                  MOVB    #RMOF,(R2)+
027522  112722  000034                  MOVB    #RMDC,(R2)+
027526  112722  000004                  MOVB    #RMBA,(R2)+
027532  112722  000002                  MOVB    #RMWC,(R2)+
027536  112722  000000                  MOVB    #RMCS1,(R2)+
027542  112712  000200                  MOVB    #200,(R2)

                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
```

```
7313  027546  004737  042712                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7314
7315  027552  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7316  027556  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
7317  027560  000240                          NOP                     ;RETURN HERE IF ERROR
7318  027562  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7319  027564  000137  027730                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7320  027570                          250$:
7321
7322                                  ;SEE IF DEVICE STARTED COMMAND
7323  027570  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7324  027574  000404                          BR      255$                    ;GO TO 255$ IF NO ERROR
7325  027576  000240                          NOP                     ;RETURN HERE IF ERROR
7326  027600  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7327  027602  000137  027730                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7328  027606  032737  020000  001340  255$:   BIT     #PIP,RMDSI              ;WAS COMMAND EXECUTED??
7329  027614  001414                          BEQ     256$                    ;NO!!
7330  027616  013737  001340  001140          MOV     RMDSI,$GDDAT            ;EXPECTED STATUS
7331  027624  042737  020000  001140          BIC     #PIP,$GDDAT
7332  027632  013737  001340  001142          MOV     RMDSI,$BDDAT            ;RECEIVED STATUS
7333  027640  104347                          ERROR   347                     ;NO ABORT
7334  027642  000137  027730                  JMP     350$
7335  027646                          256$:
7336
7337                                  ;WAIT FOR READ OPERATION TO COMPLETE
7338  027646  004737  043606                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7339
7340                                  ;GO READ STATUS FOR READ OPERATION
7341  027652  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7342  027656  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
7343  027660  000240                          NOP                     ;RETURN HERE IF ERROR
7344  027662  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7345  027664  000137  027730                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7346  027670                          260$:
7347
7348                                  ;CHECK FOR ERRORS DURING READ OPERATION
7349  027670  004737  043772                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
7350  027674  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
7351  027676  000240                          NOP                     ;RETURN HERE IF ERROR
7352  027700  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7353  027702  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7354  027704  000137  027730                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7355  027710                          270$:
7356  027710  004737  044624                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
7357  027714  000405                          BR      300$                    ;GO TO 300$ IF NO ERROR
7358  027716  000240                          NOP                     ;RETURN HERE IF ERROR
7359  027720  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7360  027722  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7361  027724  000137  027730                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7362  027730                          300$:
7363
7364  027730                          350$:
7365                                  ;;*****************************************************************
7366                                  ;*TEST 22       WRITE, READ W/ BAI
7367                                  ;;*****************************************************************
7368  027730                          TST22:
```

```
7369  027730  000004                          SCOPE                   ;SCOPE CALL
7370  027732  000240                          NOP                     ;START OF TEST
7371  027734  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7372  027740  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7373  027744  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7374  027750  012737  000022  001226          MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7375
7376                                  ;;*****************************************************************
7377                                  ;LOOP #1        FORMAT,WRITE,READ
7378
7379  027756                          10$:
7380
7381                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
7382  027756  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7383  027762  054130                          .WORD   054130  ;TASK DESCRIPTOR
7384  027764  000404                          BR      20$                     ;GO TO 20$ IF NO ERROR
7385  027766  000240                          NOP                     ;RETURN HERE IF ERROR
7386  027770  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7387  027772  000137  031006                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7388  027776                          20$:
7389
7390                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
7391  027776  012737  000000  001432          MOV     #0,RMDC0                ;CYLINDER = 0
7392  030004  012737  000000  001404          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
7393  030012  012737  106520  001402          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
7394  030020  012737  177376  001400          MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
7395  030026  012737  010000  001430          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7396  030034  012737  000063  001376          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
7397
7398                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
7399
7400  030042  004737  040174                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
7401  030046  000405                          BR      25$                     ;GO TO 25$ IF NO ERROR
7402  030050  104401  066704                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
7403  030054  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
7404  030056  000137  031006                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7405  030062                          25$:
7406  030062  012737  070016  001174          MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
7407  030070  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
7408  030076  004737  042100                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
7409
7410                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
7411  030102  012702  001533                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
7412  030106  112722  000034                  MOVB    #RMDC,(R2)+
7413  030112  112722  000006                  MOVB    #RMDA,(R2)+
7414  030116  112722  000004                  MOVB    #RMBA,(R2)+
7415  030122  112722  000002                  MOVB    #RMWC,(R2)+
7416  030126  112722  000032                  MOVB    #RMOF,(R2)+
7417  030132  112722  000000                  MOVB    #RMCS1,(R2)+
7418  030136  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
7419  030142                          30$:
7420
7421                                  ;FORMAT THE DRIVE
7422  030142  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7423  030146  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
7424  030150  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
7425  030152  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7426  030154  000137  031006                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7427  030160                        40$:
7428
7429                                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
7430  030160  004737  042712                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7431
7432                                ;WAIT FOR THE FORMAT TO COMPLETE
7433  030164  004737  043606                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7434
7435                                ;GO READ STATUS FOR FORMAT OPERATION
7436  030170  004737  042776              JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7437  030174  000404                      BR      50$                     ;GO TO 50$ IF NO ERROR
7438  030176  000240                      NOP                     ;RETURN HERE IF ERROR
7439  030200  104000                      ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7440  030202  000137  031006              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7441  030206                        50$:
7442
7443                                ;VERIFY NO ERRORS DURING FORMAT
7444  030206  004737  056276              JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7445  030212  000405                      BR      60$                     ;GO TO 60$ IF NO ERROR
7446  030214  000240                      NOP                     ;RETURN HERE IF ERROR
7447  030216  104000                      ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7448  030220  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7449  030222  000137  031006              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7450  030226                        60$:
7451
7452                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
7453  030226  012737  030244  001122        MOV     #70$,$LPADR
7454  030234  012737  030244  001124        MOV     #70$,$LPERR
7455  030242  000410                      BR      80$                     ;SKIP TO WRITE OPERATION
7456
7457                                ;;*****************************************************************
7458                                ;LOOP #2        WRITE,READ
7459
7460  030244                        70$:
7461
7462
7463                                ;PREPARE DEVICE FOR WRITE OPERATION
7464  030244  004737  037260              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7465  030250  054130                      .WORD   054130  ;TASK DESCRIPTOR
7466  030252  000404                      BR      80$                     ;GO TO 80$ IF NO ERROR
7467  030254  000240                      NOP                     ;RETURN HERE IF ERROR
7468  030256  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7469  030260  000137  031006              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7470  030264                        80$:
7471
7472
7473  030264                        120$:
7474
7475                                ;WRITE DATA TO THE DRIVE
7476  030264  111102                      MOVB    (R1),R2                 ;GENERATE CS2
7477  030266  042702  177770              BIC     #^C<U0!U1!U2>,R2
7478  030272  052702  000010              BIS     #BAI,R2
7479  030276  010237  001406              MOV     R2,RMCS20               ;INHIBIT ADDRESS INCREMENT
7480  030302  012737  177400  001400        MOV     #<^C256.+1>,RMWC0       ;CHANGE WORD COUNT
```

```
7481  030310  012737  106524  001402          MOV     #BUFONE+4,RMBA0          ;CHANGE ADDRESS
7482  030316  012737  000061  001376          MOV     #WD!GO,RMCS10            ;WRITE DATA COMMAND
7483  030324  012702  001533                  MOV     #PUTINX,R2               ;LOAD PUT REGISTER INDEX TABLE
7484  030330  112722  000010                  MOVB    #RMCS2,(R2)+
7485  030334  112722  000006                  MOVB    #RMDA,(R2)+
7486  030340  112722  000034                  MOVB    #RMDC,(R2)+
7487  030344  112722  000032                  MOVB    #RMOF,(R2)+
7488  030350  112722  000004                  MOVB    #RMBA,(R2)+
7489  030354  112722  000002                  MOVB    #RMWC,(R2)+
7490  030360  112722  000000                  MOVB    #RMCS1,(R2)+
7491  030364  112722  000200                  MOVB    #200,(R2)+               ;TERMINATE TABLE
7492
7493  030370  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7494  030374  000404                          BR      130$                     ;GO TO 130$ IF NO ERROR
7495  030376  000240                          NOP                      ;RETURN HERE IF ERROR
7496  030400  104000                          ERROR                    ;ERROR # DEFINED BY PUT SUBROUTINE
7497  030402  000137  031006                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
7498  030406                          130$:
7499
7500                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7501  030406  004737  042712                  JSR     PC,GETSTS        ;GO TO GETSTS SUBROUTINE
7502
7503                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
7504  030412  004737  043606                  JSR     PC,TIMOUT        ;GO TO TIMOUT SUBROUTINE
7505
7506                                  ;GO READ STATUS FOR WRITE COMMAND
7507  030416  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7508  030422  000404                          BR      140$                     ;GO TO 140$ IF NO ERROR
7509  030424  000240                          NOP                      ;RETURN HERE IF ERROR
7510  030426  104000                          ERROR                    ;ERROR # DEFINED BY GET SUBROUTINE
7511  030430  000137  031006                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
7512  030434                          140$:
7513
7514                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
7515  030434  004737  043772                  JSR     PC,PRIERR        ;GO CHECK FOR PRIMARY ERRORS
7516  030440  000405                          BR      150$                     ;GO TO 150$ IF NO ERROR
7517  030442  000240                          NOP                      ;RETURN HERE IF ERROR
7518  030444  104000                          ERROR                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
7519  030446  004736                          JSR     PC,@(SP)+        ;GO BACK FOR MORE ERROR CHECKS
7520  030450  000137  031006                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
7521  030454                          150$:
7522  030454  004737  056276                  JSR     PC,DTASTS        ;GO VERIFY RESULTS OF DATA TRANSFER
7523  030460  000405                          BR      160$                     ;GO TO 160$ IF NO ERROR
7524  030462  000240                          NOP                      ;RETURN HERE IF ERROR
7525  030464  104000                          ERROR                    ;ERROR # DEFINED BY DTASTS SUBROUTINE
7526  030466  004736                          JSR     PC,@(SP)+        ;GO BACK FOR MORE ERROR CHECKS
7527  030470  000137  031006                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
7528  030474                          160$:
7529
7530  030474                          170$:
7531  030474  004737  044624                  JSR     PC,SECERR        ;GO CHECK FOR SECONDARY ERRORS
7532  030500  000405                          BR      180$                     ;GO TO 180$ IF NO ERROR
7533  030502  000240                          NOP                      ;RETURN HERE IF ERROR
7534  030504  104000                          ERROR                    ;ERROR # DEFINED BY SECERR SUBROUTINE
7535  030506  004736                          JSR     PC,@(SP)+        ;GO BACK FOR MORE ERROR CHECKS
7536  030510  000137  031006                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
```

```
7537  030514                                  180$:
7538
7539                                          ;CHANGE LOOP ADDRESSES
7540  030514  012737  030532  001122                  MOV     #190$,$LPADR
7541  030522  012737  030532  001124                  MOV     #190$,$LPERR
7542  030530  000410                                  BR      200$                    ;SKIP TO NEXT OPERATION
7543
7544                                          ;;*****************************************************************
7545                                          ;LOOP #3        READ
7546
7547  030532                                  190$:
7548
7549                                          ;PREPARE DEVICE FOR READ OPERATION
7550  030532  004737  037260                          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7551  030536  054130                                  .WORD   054130  ;TASK DESCRIPTOR
7552  030540  000404                                  BR      200$                    ;GO TO 200$ IF NO ERROR
7553  030542  000240                                  NOP                     ;RETURN HERE IF ERROR
7554  030544  104000                                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7555  030546  000137  031006                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7556  030552                                  200$:
7557
7558  030552                                  240$:
7559
7560                                          ;READ DATA FROM DEVICE
7561  030552  111102                                  MOVB    (R1),R2                 ;RESET BAI
7562  030554  042702  177770                          BIC     #^C<U0!U1!U2>,R2
7563  030560  010237  001406                          MOV     R2,RMCS20
7564  030564  012737  107530  001402                  MOV     #BUFTWO+4,RMBA0         ;CHANGE BUFFER
7565  030572  012737  000071  001376                  MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
7566  030600  012702  001533                          MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7567  030604  112722  000006                          MOVB    #RMDA,(R2)+
7568  030610  112722  000032                          MOVB    #RMOF,(R2)+
7569  030614  112722  000034                          MOVB    #RMDC,(R2)+
7570  030620  112722  000004                          MOVB    #RMBA,(R2)+
7571  030624  112722  000002                          MOVB    #RMWC,(R2)+
7572  030630  112722  000000                          MOVB    #RMCS1,(R2)+
7573  030634  112712  000200                          MOVB    #200,(R2)
7574
7575  030640  004737  043246                          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7576  030644  000404                                  BR      250$                    ;GO TO 250$ IF NO ERROR
7577  030646  000240                                  NOP                     ;RETURN HERE IF ERROR
7578  030650  104000                                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7579  030652  000137  031006                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7580  030656                                  250$:
7581
7582                                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7583  030656  004737  042712                          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7584
7585                                          ;WAIT FOR READ OPERATION TO COMPLETE
7586  030662  004737  043606                          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7587
7588                                          ;GO READ STATUS FOR READ OPERATION
7589  030666  004737  042776                          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7590  030672  000404                                  BR      260$                    ;GO TO 260$ IF NO ERROR
7591  030674  000240                                  NOP                     ;RETURN HERE IF ERROR
7592  030676  104000                                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
```

```
7593  030700  000137  031006                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7594  030704                            260$:
7595
7596                                    ;CHECK FOR ERRORS DURING READ OPERATION
7597  030704  004737  043772                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7598  030710  000405                        BR      270$                    ;GO TO 270$ IF NO ERROR
7599  030712  000240                        NOP                     ;RETURN HERE IF ERROR
7600  030714  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7601  030716  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7602  030720  000137  031006                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7603  030724                            270$:
7604  030724  004737  056276                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7605  030730  000405                        BR      280$                    ;GO TO 280$ IF NO ERROR
7606  030732  000240                        NOP                     ;RETURN HERE IF ERROR
7607  030734  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7608  030736  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7609  030740  000137  031006                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7610  030744                            280$:
7611
7612  030744                            290$:
7613  030744  004737  044624                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7614  030750  000405                        BR      300$                    ;GO TO 300$ IF NO ERROR
7615  030752  000240                        NOP                     ;RETURN HERE IF ERROR
7616  030754  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7617  030756  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7618  030760  000137  031006                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7619  030764                            300$:
7620  030764  004737  042344                JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
7621  030770  106524                        .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
7622  030772  107530                        .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
7623  030774  000404                        BR      310$                    ;GO TO 310$ IF NO ERROR
7624  030776  000240                        NOP                     ;RETURN HERE IF ERROR
7625  031000  104000                        ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
7626  031002  000137  031006                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7627  031006                            310$:
7628
7629  031006                            350$:
7630                                    ;;*****************************************************************
7631                                    ;*TEST 23        WRITE, READ EACH CURRENT LEVEL
7632                                    ;;*****************************************************************
7633  031006                            TST23:
7634  031006  000004                        SCOPE                   ;SCOPE CALL
7635  031010  000240                        NOP                     ;START OF TEST
7636  031012  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7637  031016  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7638  031022  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7639  031026  012737  000023  001226        MOV     #23,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7640
7641                                    ;;*****************************************************************
7642                                    ;LOOP #1        FORMAT,WRITE,READ
7643
7644  031034                            10$:
7645
7646                                    ;PREPARE THE DEVICE FOR FORMAT OPERATION
7647  031034  004737  037260                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7648  031040  054130                        .WORD   054130  ;TASK DESCRIPTOR
```

```
                        BR      20$                     ;GO TO 20$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
20$:

                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
                25$:    MOV     #0,RMDC0                ;CYLINDER = 0
                        MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
                        MOV     #BUFONE,RMBA0           ;BUS ADDRESS
                        MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
                        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
                        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND

                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE

                        JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
                        BR      26$                     ;GO TO 26$ IF NO ERROR
                        TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
                        ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
26$:
                        MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
                        MOV     #1,$TMP1                ;RANGE OF PATTERN
                        JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT

                ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
                        MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
                        MOVB    #RMDC,(R2)+
                        MOVB    #RMDA,(R2)+
                        MOVB    #RMBA,(R2)+
                        MOVB    #RMWC,(R2)+
                        MOVB    #RMOF,(R2)+
                        MOVB    #RMCS1,(R2)+
                        MOVB    #200,(R2)               ;TERMINATE TABLE
                30$:

                ;FORMAT THE DRIVE
                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
                        BR      40$                     ;GO TO 40$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                40$:

                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                ;WAIT FOR THE FORMAT TO COMPLETE
                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                ;GO READ STATUS FOR FORMAT OPERATION
                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
                        BR      50$                     ;GO TO 50$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
```

```
                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
031264                  50$:

                        ;VERIFY NO ERRORS DURING FORMAT
031264  004737  056276          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
031270  000405                  BR      60$                     ;GO TO 60$ IF NO ERROR
031272  000240                  NOP                     ;RETURN HERE IF ERROR
031274  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
031276  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
031300  000137  032060          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
031304                  60$:

                        ;MOVE LOOP ADDRESSES TO NEXT OPERATION
031304  012737  031322  001122          MOV     #70$,$LPADR
031312  012737  031322  001124          MOV     #70$,$LPERR
031320  000410                  BR      80$                     ;SKIP TO WRITE OPERATION

                        ;;*****************************************************************
                        ;LOOP #2        WRITE,READ

031322                  70$:

                        ;PREPARE DEVICE FOR WRITE OPERATION
031322  004737  037260          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
031326  054130                  .WORD   054130  ;TASK DESCRIPTOR
031330  000404                  BR      80$                     ;GO TO 80$ IF NO ERROR
031332  000240                  NOP                     ;RETURN HERE IF ERROR
031334  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
031336  000137  032060          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
031342                  80$:

031342                  120$:

                        ;WRITE DATA TO THE DRIVE
031342  012737  177400  001400          MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
031350  012737  106524  001402          MOV     #BUFONE+4,RMBAO         ;CHANGE ADDRESS
031356  012737  000061  001376          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
031364  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
031370  112722  000006                  MOVB    #RMDA,(R2)+
031374  112722  000034                  MOVB    #RMDC,(R2)+
031400  112722  000032                  MOVB    #RMOF,(R2)+
031404  112722  000004                  MOVB    #RMBA,(R2)+
031410  112722  000002                  MOVB    #RMWC,(R2)+
031414  112722  000000                  MOVB    #RMCS1,(R2)+
031420  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE

031424  004737  043246          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
031430  000404                  BR      130$                    ;GO TO 130$ IF NO ERROR
031432  000240                  NOP                     ;RETURN HERE IF ERROR
031434  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
031436  000137  032060          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
031442                  130$:

                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
```

```
7761  031442  004737  042712                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7762
7763                                    ;WAIT FOR WRITE COMMAND TO COMPLETE
7764  031446  004737  043606                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7765
7766                                    ;GO READ STATUS FOR WRITE COMMAND
7767  031452  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7768  031456  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
7769  031460  000240                          NOP                     ;RETURN HERE IF ERROR
7770  031462  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7771  031464  000137  032060                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7772  031470                          140$:
7773
7774                                    ;CHECK FOR ERRORS DURING WRITE OPERATION
7775  031470  004737  043772                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7776  031474  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
7777  031476  000240                          NOP                     ;RETURN HERE IF ERROR
7778  031500  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7779  031502  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7780  031504  000137  032060                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7781  031510                          150$:
7782  031510  004737  056276                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7783  031514  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
7784  031516  000240                          NOP                     ;RETURN HERE IF ERROR
7785  031520  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7786  031522  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7787  031524  000137  032060                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7788  031530                          160$:
7789
7790  031530                          170$:
7791  031530  004737  044624                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7792  031534  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
7793  031536  000240                          NOP                     ;RETURN HERE IF ERROR
7794  031540  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7795  031542  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7796  031544  000137  032060                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7797  031550                          180$:
7798
7799                                    ;CHANGE LOOP ADDRESSES
7800  031550  012737  031566  001122          MOV     #190$,$LPADR
7801  031556  012737  031566  001124          MOV     #190$,$LPERR
7802  031564  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
7803
7804                                    ;;*****************************************************************
7805                                    ;LOOP #3        READ
7806
7807  031566                          190$:
7808
7809                                    ;PREPARE DEVICE FOR READ OPERATION
7810  031566  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7811  031572  054130                          .WORD   054130  ;TASK DESCRIPTOR
7812  031574  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
7813  031576  000240                          NOP                     ;RETURN HERE IF ERROR
7814  031600  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7815  031602  000137  032060                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7816  031606                          200$:
```

```
7817
7818  031606                                  240$:
7819
7820                                          ;READ DATA FROM DEVICE
7821  031606  012737  107530  001402                MOV     #BUFTWO+4,RMBAO         ;CHANGE ADDRESS
7822  031614  012737  177400  001400                MOV     #<^C256.+1>,RMWCO
7823  031622  012737  000071  001376                MOV     #RD!GO,RMCS1O           ;READ DATA COMMAND
7824  031630  012702  001533                        MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7825  031634  112722  000006                        MOVB    #RMDA,(R2)+
7826  031640  112722  000032                        MOVB    #RMOF,(R2)+
7827  031644  112722  000034                        MOVB    #RMDC,(R2)+
7828  031650  112722  000004                        MOVB    #RMBA,(R2)+
7829  031654  112722  000002                        MOVB    #RMWC,(R2)+
7830  031660  112722  000000                        MOVB    #RMCS1,(R2)+
7831  031664  112712  000200                        MOVB    #200,(R2)
7832
7833  031670  004737  043246                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7834  031674  000404                                BR      250$                    ;GO TO 250$ IF NO ERROR
7835  031676  000240                                NOP                     ;RETURN HERE IF ERROR
7836  031700  104000                                ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7837  031702  000137  032060                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7838  031706                                  250$:
7839
7840                                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7841  031706  004737  042712                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7842
7843                                          ;WAIT FOR READ OPERATION TO COMPLETE
7844  031712  004737  043606                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7845
7846                                          ;GO READ STATUS FOR READ OPERATION
7847  031716  004737  042776                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7848  031722  000404                                BR      260$                    ;GO TO 260$ IF NO ERROR
7849  031724  000240                                NOP                     ;RETURN HERE IF ERROR
7850  031726  104000                                ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7851  031730  000137  032060                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7852  031734                                  260$:
7853
7854                                          ;CHECK FOR ERRORS DURING READ OPERATION
7855  031734  004737  043772                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7856  031740  000405                                BR      270$                    ;GO TO 270$ IF NO ERROR
7857  031742  000240                                NOP                     ;RETURN HERE IF ERROR
7858  031744  104000                                ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7859  031746  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7860  031750  000137  032060                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7861  031754                                  270$:
7862  031754  004737  056276                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7863  031760  000405                                BR      280$                    ;GO TO 280$ IF NO ERROR
7864  031762  000240                                NOP                     ;RETURN HERE IF ERROR
7865  031764  104000                                ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7866  031766  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7867  031770  000137  032060                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7868  031774                                  280$:
7869
7870  031774                                  290$:
7871  031774  004737  044624                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7872  032000  000405                                BR      300$                    ;GO TO 300$ IF NO ERROR
```

```
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                300$:
                                        JSR     PC,CMPBUF               ;GO COMPARE WRITE, READ DATA BUFFERS
                                        .WORD   BUFONE+4                        ;STARTING ADDRESS OF WRITE BUFFER
                                        .WORD   BUFTWO+4                        ;STARTING ADDDRESS OF READ BUFFER
                                        BR      310$                            ;GO TO 310$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY CMPBUF SUBROUTINE
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                310$:
                                        ADD     #128.,RMDC0                     ;ADVANCE TO NEXT THRESHOLD
                                        CMP     RMDC0,#768.                     ;DONE??
                                        BHI     350$                            ;YES!!
                                        JMP     25$                             ;DO NEXT CURRENT LEVEL

                                350$:
                                        MOV     #10$,$LPADR                     ;LOOP BACK TO START OF TEST
                                        MOV     #10$,$LPERR
                                ;;*****************************************************************
                                ;*TEST 24        WRITE, WRITE CHECK W/ CURRENT LEVEL SWITCHING
                                ;;*****************************************************************
                                TST24:
                                        SCOPE                           ;SCOPE CALL
                                        NOP                             ;START OF TEST
                                        MOV     #STACK,SP               ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
                                        MOV     #24,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX

                                ;;*****************************************************************
                                ;LOOP #1        FORMAT,WRITE,READ

                                10$:

                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
                                        JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
                                        .WORD   054130  ;TASK DESCRIPTOR
                                        BR      20$                             ;GO TO 20$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                20$:

                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
                                        MOV     #383.,RMDC0                     ;CYLINDER = 383.
                                        MOV     #2037,RMDA0                     ;TRACK = 4, SECTOR = 31
                                        MOV     #BUFONE,RMBA0                   ;BUS ADDRESS
                                        MOV     #^C<2*<256.+2>>+1>,RMWC0                ;WORD COUNT
                                        MOV     #FMT16,RMOF0                    ;16 BIT FORMAT
                                        MOV     #WH!GO,RMCS10                   ;WRITE HEADER AND DATA COMMAND

                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
```

```
7929  032206  004737  040174            JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
7930  032212  000405                      BR      25$                     ;GO TO 25$ IF NO ERROR
7931  032214  104401  066704            TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
7932  032220  104000                      ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
7933  032222  000137  033066            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7934  032226                          25$:
7935  032226  012737  070016  001174      MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
7936  032234  012737  000001  001176      MOV     #1,$TMP1                ;RANGE OF PATTERN
7937  032242  004737  042100            JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
7938
7939                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
7940  032246  012702  001533            MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
7941  032252  112722  000034            MOVB    #RMDC,(R2)+
7942  032256  112722  000006            MOVB    #RMDA,(R2)+
7943  032262  112722  000004            MOVB    #RMBA,(R2)+
7944  032266  112722  000002            MOVB    #RMWC,(R2)+
7945  032272  112722  000032            MOVB    #RMOF,(R2)+
7946  032276  112722  000000            MOVB    #RMCS1,(R2)+
7947  032302  112712  000200            MOVB    #200,(R2)               ;TERMINATE TABLE
7948  032306                          30$:
7949
7950                                  ;FORMAT THE DRIVE
7951  032306  004737  043246            JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7952  032312  000404                      BR      40$                     ;GO TO 40$ IF NO ERROR
7953  032314  000240                      NOP                     ;RETURN HERE IF ERROR
7954  032316  104000                      ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7955  032320  000137  033066            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7956  032324                          40$:
7957
7958                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
7959  032324  004737  042712            JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7960
7961                                  ;WAIT FOR THE FORMAT TO COMPLETE
7962  032330  004737  043606            JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7963
7964                                  ;GO READ STATUS FOR FORMAT OPERATION
7965  032334  004737  042776            JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7966  032340  000404                      BR      50$                     ;GO TO 50$ IF NO ERROR
7967  032342  000240                      NOP                     ;RETURN HERE IF ERROR
7968  032344  104000                      ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7969  032346  000137  033066            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7970  032352                          50$:
7971
7972                                  ;VERIFY NO ERRORS DURING FORMAT
7973  032352  004737  056276            JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7974  032356  000405                      BR      60$                     ;GO TO 60$ IF NO ERROR
7975  032360  000240                      NOP                     ;RETURN HERE IF ERROR
7976  032362  104000                      ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7977  032364  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7978  032366  000137  033066            JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7979  032372                          60$:
7980
7981                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
7982  032372  012737  032410  001122      MOV     #70$,$LPADR
7983  032400  012737  032410  001124      MOV     #70$,$LPERR
7984  032406  000410                      BR      80$                     ;SKIP TO WRITE OPERATION
```

```
                                ;;*****************************************************************
                                ;LOOP #2        WRITE,READ

032410                          70$:

                                ;PREPARE DEVICE FOR WRITE OPERATION
032410  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
032414  054130                          .WORD   054130  ;TASK DESCRIPTOR
032416  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
032420  000240                          NOP                     ;RETURN HERE IF ERROR
032422  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
032424  000137  033066                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
032430                          80$:

032430                          120$:

                                ;WRITE DATA TO THE DRIVE
032430  012737  177000  001400          MOV     #<^C<2*256.>+1>,RMWC0   ;CHANGE WORD COUNT
032436  012737  106524  001402          MOV     #BUFONE+4,RMBA0         ;CHANGE ADDRESS
032444  012737  000061  001376          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
032452  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
032456  112722  000006                  MOVB    #RMDA,(R2)+
032462  112722  000034                  MOVB    #RMDC,(R2)+
032466  112722  000032                  MOVB    #RMOF,(R2)+
032472  112722  000004                  MOVB    #RMBA,(R2)+
032476  112722  000002                  MOVB    #RMWC,(R2)+
032502  112722  000000                  MOVB    #RMCS1,(R2)+
032506  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE

032512  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
032516  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
032520  000240                          NOP                     ;RETURN HERE IF ERROR
032522  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
032524  000137  033066                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
032530                          130$:

                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
032530  004737  042712                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR WRITE COMMAND TO COMPLETE
032534  004737  043606                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR WRITE COMMAND
032540  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
032544  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
032546  000240                          NOP                     ;RETURN HERE IF ERROR
032550  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
032552  000137  033066                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
032556                          140$:

                                ;CHECK FOR ERRORS DURING WRITE OPERATION
032556  004737  043772                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
032562  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
```

```
8041  032564  000240                          NOP                             ;RETURN HERE IF ERROR
8042  032566  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
8043  032570  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8044  032572  000137  033066                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
8045  032576                          150$:
8046  032576  004737  056276                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
8047  032602  000405                          BR      160$                            ;GO TO 160$ IF NO ERROR
8048  032604  000240                          NOP                             ;RETURN HERE IF ERROR
8049  032606  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
8050  032610  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8051  032612  000137  033066                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
8052  032616                          160$:
8053
8054  032616                          170$:
8055  032616  004737  044624                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
8056  032622  000405                          BR      180$                            ;GO TO 180$ IF NO ERROR
8057  032624  000240                          NOP                             ;RETURN HERE IF ERROR
8058  032626  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
8059  032630  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8060  032632  000137  033066                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
8061  032636                          180$:
8062
8063                                  ;CHANGE LOOP ADDRESSES
8064  032636  012737  032654  001122          MOV     #190$,$LPADR
8065  032644  012737  032654  001124          MOV     #190$,$LPERR
8066  032652  000410                          BR      200$                            ;SKIP TO NEXT OPERATION
8067
8068                                  ;;*****************************************************************
8069                                  ;LOOP #3        READ
8070
8071  032654                          190$:
8072
8073                                  ;PREPARE DEVICE FOR READ OPERATION
8074  032654  004737  037260                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
8075  032660  054130                          .WORD   054130  ;TASK DESCRIPTOR
8076  032662  000404                          BR      200$                            ;GO TO 200$ IF NO ERROR
8077  032664  000240                          NOP                             ;RETURN HERE IF ERROR
8078  032666  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8079  032670  000137  033066                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
8080  032674                          200$:
8081
8082  032674                          240$:
8083
8084                                  ;READ DATA FROM DEVICE
8085  032674  012737  000051  001376          MOV     #WCO!GO,RMCS10                  ;READ DATA COMMAND
8086  032702  012702  001533                  MOV     #PUTINX,R2                      ;LOAD PUT REGISTER INDEX TABLE
8087  032706  112722  000006                  MOVB    #RMDA,(R2)+
8088  032712  112722  000032                  MOVB    #RMOF,(R2)+
8089  032716  112722  000034                  MOVB    #RMDC,(R2)+
8090  032722  112722  000004                  MOVB    #RMBA,(R2)+
8091  032726  112722  000002                  MOVB    #RMWC,(R2)+
8092  032732  112722  000000                  MOVB    #RMCS1,(R2)+
8093  032736  112712  000200                  MOVB    #200,(R2)
8094
8095  032742  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8096  032746  000404                          BR      250$                            ;GO TO 250$ IF NO ERROR
```

```
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY PUT SUBROUTINE
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                250$:

                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
                                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR READ OPERATION TO COMPLETE
                                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR READ OPERATION
                                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
                                        BR      260$                    ;GO TO 260$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY GET SUBROUTINE
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                260$:

                                ;CHECK FOR ERRORS DURING READ OPERATION
                                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                                        BR      270$                    ;GO TO 270$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                270$:
                                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
                                        BR      280$                    ;GO TO 280$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                280$:

                                290$:
                                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                                        BR      300$                    ;GO TO 300$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                300$:

                                350$:
                                ;;*****************************************************************
                                ;*TEST 25       WRITE, READ EARLY PEAK SHIFT
                                ;;*****************************************************************
                                TST25:
                                        SCOPE                           ;SCOPE CALL
                                        NOP                             ;START OF TEST
                                        MOV     #STACK,SP               ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
                                        MOV     #25,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
```

```
                        ;;*****************************************************************
                        ;LOOP #1        FORMAT,WRITE,READ

033114                  10$:

                        ;PREPARE THE DEVICE FOR FORMAT OPERATION
033114  004737  037260          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
033120  054130                  .WORD   054130  ;TASK DESCRIPTOR
033122  000404                  BR      20$                     ;GO TO 20$ IF NO ERROR
033124  000240                  NOP                     ;RETURN HERE IF ERROR
033126  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
033130  000137  034110          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
033134                  20$:

                        ;LOAD PARAMETERS AND GENERATE DATA BUFFER
033134  012737  000000  001432          MOV     #0,RMDC0                ;CYLINDER = 0
033142  012737  000000  001404          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
033150  012737  106520  001402          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
033156  012737  177376  001400          MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
033164  012737  010000  001430          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
033172  012737  000063  001376          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND

                        ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE

033200  004737  040174          JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
033204  000405                  BR      25$                     ;GO TO 25$ IF NO ERROR
033206  104401  066704          TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
033212  104000                  ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
033214  000137  034110          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
033220                  25$:
033220  012737  070272  001174          MOV     #EARLY,$TMP0            ;STARTING ADDRESS OF PATTERN
033226  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
033234  004737  042100          JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT

                        ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
033240  012702  001533          MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
033244  112722  000034          MOVB    #RMDC,(R2)+
033250  112722  000006          MOVB    #RMDA,(R2)+
033254  112722  000004          MOVB    #RMBA,(R2)+
033260  112722  000002          MOVB    #RMWC,(R2)+
033264  112722  000032          MOVB    #RMOF,(R2)+
033270  112722  000000          MOVB    #RMCS1,(R2)+
033274  112712  000200          MOVB    #200,(R2)               ;TERMINATE TABLE
033300                  30$:

                        ;FORMAT THE DRIVE
033300  004737  043246          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
033304  000404                  BR      40$                     ;GO TO 40$ IF NO ERROR
033306  000240                  NOP                     ;RETURN HERE IF ERROR
033310  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
033312  000137  034110          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
033316                  40$:

                        ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
033316  004737  042712          JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
```

```
                                ;WAIT FOR THE FORMAT TO COMPLETE
033322  004737  043606                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR FORMAT OPERATION
033326  004737  042776          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
033332  000404                  BR      50$                     ;GO TO 50$ IF NO ERROR
033334  000240                  NOP                     ;RETURN HERE IF ERROR
033336  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
033340  000137  034110          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
033344                          50$:

                                ;VERIFY NO ERRORS DURING FORMAT
033344  004737  056276          JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
033350  000405                  BR      60$                     ;GO TO 60$ IF NO ERROR
033352  000240                  NOP                     ;RETURN HERE IF ERROR
033354  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
033356  004736                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
033360  000137  034110          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
033364                          60$:

                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
033364  012737  033402  001122          MOV     #70$,$LPADR
033372  012737  033402  001124          MOV     #70$,$LPERR
033400  000410                  BR      80$                     ;SKIP TO WRITE OPERATION

                                ;;*****************************************************************
                                ;LOOP #2        WRITE,READ

033402                          70$:

                                ;PREPARE DEVICE FOR WRITE OPERATION
033402  004737  037260          JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
033406  054130                  .WORD   054130  ;TASK DESCRIPTOR
033410  000404                  BR      80$                     ;GO TO 80$ IF NO ERROR
033412  000240                  NOP                     ;RETURN HERE IF ERROR
033414  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
033416  000137  034110          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
033422                          80$:

033422                          120$:

                                ;WRITE DATA TO THE DRIVE
033422  012737  177400  001400          MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
033430  012737  106524  001402          MOV     #BUFONE+4,RMBAO         ;CHANGE ADDRESS
033436  012737  003061  001376          MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
033444  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
033450  112722  000006                  MOVB    #RMDA,(R2)+
033454  112722  000034                  MOVB    #RMDC,(R2)+
033460  112722  000032                  MOVB    #RMOF,(R2)+
033464  112722  000004                  MOVB    #RMBA,(R2)+
033470  112722  000002                  MOVB    #RMWC,(R2)+
033474  112722  000000                  MOVB    #RMCS1,(R2)+
033500  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
```

```
 8265  033504  004737  043246                 JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
 8266  033510  000404                         BR      130$                    ;GO TO 130$ IF NO ERROR
 8267  033512  000240                         NOP                     ;RETURN HERE IF ERROR
 8268  033514  104000                         ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
 8269  033516  000137  034110                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8270  033522                         130$:
 8271
 8272                                 ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
 8273  033522  004737  042712                 JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
 8274
 8275                                 ;WAIT FOR WRITE COMMAND TO COMPLETE
 8276  033526  004737  043606                 JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
 8277
 8278                                 ;GO READ STATUS FOR WRITE COMMAND
 8279  033532  004737  042776                 JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 8280  033536  000404                         BR      140$                    ;GO TO 140$ IF NO ERROR
 8281  033540  000240                         NOP                     ;RETURN HERE IF ERROR
 8282  033542  104000                         ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 8283  033544  000137  034110                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8284  033550                         140$:
 8285
 8286                                 ;CHECK FOR ERRORS DURING WRITE OPERATION
 8287  033550  004737  043772                 JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 8288  033554  000405                         BR      150$                    ;GO TO 150$ IF NO ERROR
 8289  033556  000240                         NOP                     ;RETURN HERE IF ERROR
 8290  033560  104000                         ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 8291  033562  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8292  033564  000137  034110                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8293  033570                         150$:
 8294  033570  004737  056276                 JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 8295  033574  000405                         BR      160$                    ;GO TO 160$ IF NO ERROR
 8296  033576  000240                         NOP                     ;RETURN HERE IF ERROR
 8297  033600  104000                         ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 8298  033602  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8299  033604  000137  034110                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8300  033610                         160$:
 8301
 8302  033610                         170$:
 8303  033610  004737  044624                 JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 8304  033614  000405                         BR      180$                    ;GO TO 180$ IF NO ERROR
 8305  033616  000240                         NOP                     ;RETURN HERE IF ERROR
 8306  033620  104000                         ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 8307  033622  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8308  033624  000137  034110                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8309  033630                         180$:
 8310
 8311                                 ;CHANGE LOOP ADDRESSES
 8312  033630  012737  033646  001122         MOV     #190$,$LPADR
 8313  033636  012737  033646  001124         MOV     #190$,$LPERR
 8314  033644  000410                         BR      200$                    ;SKIP TO NEXT OPERATION
 8315
 8316                                 ;;*****************************************************************
 8317                                 ;LOOP #3        READ
 8318
 8319  033646                         190$:
 8320
```

```
 8321
 8322  033646  004737  037260          ;PREPARE DEVICE FOR READ OPERATION
 8323  033652  054130                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 8324  033654  000404                        .WORD   054130  ;TASK DESCRIPTOR
 8325  033656  000240                        BR      200$                    ;GO TO 200$ IF NO ERROR
 8326  033660  104000                        NOP                     ;RETURN HERE IF ERROR
 8327  033662  000137  034110                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 8328  033666                                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8329                                  200$:
 8330  033666                          240$:
 8331
 8332                                  ;READ DATA FROM DEVICE
 8333  033666  012737  000071  001376        MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
 8334  033674  012737  107530  001402        MOV     #BUFTWO+4,RMBA0         ;CHANGE BUFFER
 8335  033702  012702  001533                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
 8336  033706  112722  000006                MOVB    #RMDA,(R2)+
 8337  033712  112722  000032                MOVB    #RMOF,(R2)+
 8338  033716  112722  000034                MOVB    #RMDC,(R2)+
 8339  033722  112722  000004                MOVB    #RMBA,(R2)+
 8340  033726  112722  000002                MOVB    #RMWC,(R2)+
 8341  033732  112722  000000                MOVB    #RMCS1,(R2)+
 8342  033736  112712  000200                MOVB    #200,(R2)
 8343
 8344  033742  004737  043246                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
 8345  033746  000404                        BR      250$                    ;GO TO 250$ IF NO ERROR
 8346  033750  000240                        NOP                     ;RETURN HERE IF ERROR
 8347  033752  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
 8348  033754  000137  034110                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8349  033760                          250$:
 8350
 8351                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
 8352  033760  004737  042712                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
 8353
 8354                                  ;WAIT FOR READ OPERATION TO COMPLETE
 8355  033764  004737  043606                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
 8356
 8357                                  ;GO READ STATUS FOR READ OPERATION
 8358  033770  004737  042776                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 8359  033774  000404                        BR      260$                    ;GO TO 260$ IF NO ERROR
 8360  033776  000240                        NOP                     ;RETURN HERE IF ERROR
 8361  034000  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 8362  034002  000137  034110                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8363  034006                          260$:
 8364
 8365                                  ;CHECK FOR ERRORS DURING READ OPERATION
 8366  034006  004737  043772                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 8367  034012  000405                        BR      270$                    ;GO TO 270$ IF NO ERROR
 8368  034014  000240                        NOP                     ;RETURN HERE IF ERROR
 8369  034016  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 8370  034020  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8371  034022  000137  034110                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8372  034026                          270$:
 8373  034026  004737  056276                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 8374  034032  000405                        BR      280$                    ;GO TO 280$ IF NO ERROR
 8375  034034  000240                        NOP                     ;RETURN HERE IF ERROR
 8376  034036  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
```

```
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                280$:
                                290$:
                                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                                        BR      300$                    ;GO TO 300$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                300$:
                                        JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
                                        .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
                                        .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
                                        BR      310$                    ;GO TO 310$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                310$:
                                350$:
                                ;;*****************************************************************
                                ;*TEST 26        WRITE, READ MIXED PEAK SHIFT
                                ;;*****************************************************************
                                TST26:
                                        SCOPE                   ;SCOPE CALL
                                        NOP                     ;START OF TEST
                                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                                        MOV     #26,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                ;;*****************************************************************
                                ;LOOP #1        FORMAT,WRITE,READ
                                10$:
                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
                                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                                        .WORD   054130  ;TASK DESCRIPTOR
                                        BR      20$                     ;GO TO 20$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                20$:
                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
                                        MOV     #0,RMDC0                ;CYLINDER = 0
                                        MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
                                        MOV     #BUFONE,RMBA0           ;BUS ADDRESS
                                        MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
                                        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
                                        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
```

```
8433
8434  034222  004737  040174          JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
8435  034226  000405                  BR      25$                     ;GO TO 25$ IF NO ERROR
8436  034230  104401  066704          TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
8437  034234  104000                  ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
8438  034236  000137  035132          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8439  034242                  25$:
8440  034242  012737  067756  001174  MOV     #MIXED,$TMP0            ;STARTING ADDRESS OF PATTERN
8441  034250  012737  000200  001176  MOV     #128.,$TMP1             ;RANGE OF PATTERN
8442  034256  004737  042100          JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
8443
8444                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
8445  034262  012702  001533          MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
8446  034266  112722  000034          MOVB    #RMDC,(R2)+
8447  034272  112722  000006          MOVB    #RMDA,(R2)+
8448  034276  112722  000004          MOVB    #RMBA,(R2)+
8449  034302  112722  000002          MOVB    #RMWC,(R2)+
8450  034306  112722  000032          MOVB    #RMOF,(R2)+
8451  034312  112722  000000          MOVB    #RMCS1,(R2)+
8452  034316  112712  000200          MOVB    #200,(R2)               ;TERMINATE TABLE
8453  034322                  30$:
8454
8455                  ;FORMAT THE DRIVE
8456  034322  004737  043246          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8457  034326  000404                  BR      40$                     ;GO TO 40$ IF NO ERROR
8458  034330  000240                  NOP                     ;RETURN HERE IF ERROR
8459  034332  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8460  034334  000137  035132          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8461  034340                  40$:
8462
8463                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
8464  034340  004737  042712          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8465
8466                  ;WAIT FOR THE FORMAT TO COMPLETE
8467  034344  004737  043606          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8468
8469                  ;GO READ STATUS FOR FORMAT OPERATION
8470  034350  004737  042776          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8471  034354  000404                  BR      50$                     ;GO TO 50$ IF NO ERROR
8472  034356  000240                  NOP                     ;RETURN HERE IF ERROR
8473  034360  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8474  034362  000137  035132          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8475  034366                  50$:
8476
8477                  ;VERIFY NO ERRORS DURING FORMAT
8478  034366  004737  056276          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8479  034372  000405                  BR      60$                     ;GO TO 60$ IF NO ERROR
8480  034374  000240                  NOP                     ;RETURN HERE IF ERROR
8481  034376  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8482  034400  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8483  034402  000137  035132          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8484  034406                  60$:
8485
8486                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
8487  034406  012737  034424  001122  MOV     #70$,$LPADR
8488  034414  012737  034424  001124  MOV     #70$,$LPERR
```

```
8489  034422  000410                        BR      80$                     ;SKIP TO WRITE OPERATION
8490
8491                                        ;;*****************************************************************
8492                                        ;LOOP #2        WRITE,READ
8493
8494  034424                                70$:
8495
8496
8497                                        ;PREPARE DEVICE FOR WRITE OPERATION
8498  034424  004737  037260                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8499  034430  054130                              .WORD   054130  ;TASK DESCRIPTOR
8500  034432  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
8501  034434  000240                        NOP                     ;RETURN HERE IF ERROR
8502  034436  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8503  034440  000137  035132                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8504  034444                                80$:
8505
8506
8507  034444                                120$:
8508
8509                                        ;WRITE DATA TO THE DRIVE
8510  034444  012737  177400  001400                MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
8511  034452  012737  106524  001402                MOV     #BUFONE+4,RMBAO         ;CHANGE ADDRESS
8512  034460  012737  000061  001376                MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
8513  034466  012702  001533                      MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8514  034472  112722  000006                      MOVB    #RMDA,(R2)+
8515  034476  112722  000034                      MOVB    #RMDC,(R2)+
8516  034502  112722  000032                      MOVB    #RMOF,(R2)+
8517  034506  112722  000004                      MOVB    #RMBA,(R2)+
8518  034512  112722  000002                      MOVB    #RMWC,(R2)+
8519  034516  112722  000000                      MOVB    #RMCS1,(R2)+
8520  034522  112722  000200                      MOVB    #200,(R2)+              ;TERMINATE TABLE
8521
8522  034526  004737  043246                      JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8523  034532  000404                        BR      130$                    ;GO TO 130$ IF NO ERROR
8524  034534  000240                        NOP                     ;RETURN HERE IF ERROR
8525  034536  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8526  034540  000137  035132                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8527  034544                                130$:
8528
8529                                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8530  034544  004737  042712                      JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8531
8532                                        ;WAIT FOR WRITE COMMAND TO COMPLETE
8533  034550  004737  043606                      JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8534
8535
8536  034554  004737  042776                      JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8537  034560  000404                        BR      140$                    ;GO TO 140$ IF NO ERROR
8538  034562  000240                        NOP                     ;RETURN HERE IF ERROR
8539  034564  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8540  034566  000137  035132                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8541  034572                                140$:
8542
8543                                        ;CHECK FOR ERRORS DURING WRITE OPERATION
8544  034572  004737  043772                      JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
```

```
8545  034576  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
8546  034600  000240                          NOP                     ;RETURN HERE IF ERROR
8547  034602  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8548  034604  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8549  034606  000137  035132                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8550  034612                          150$:
8551  034612  004737  056276                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8552  034616  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
8553  034620  000240                          NOP                     ;RETURN HERE IF ERROR
8554  034622  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8555  034624  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8556  034626  000137  035132                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8557  034632                          160$:
8558
8559  034632                          170$:
8560  034632  004737  044624                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8561  034636  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
8562  034640  000240                          NOP                     ;RETURN HERE IF ERROR
8563  034642  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8564  034644  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8565  034646  000137  035132                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8566  034652                          180$:
8567
8568                                  ;CHANGE LOOP ADDRESSES
8569  034652  012737  034670  001122          MOV     #190$,$LPADR
8570  034660  012737  034670  001124          MOV     #190$,$LPERR
8571  034666  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
8572
8573                                  ;;*****************************************************************
8574                                  ;LOOP #3        READ
8575
8576  034670                          190$:
8577
8578                                  ;PREPARE DEVICE FOR READ OPERATION
8579  034670  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8580  034674  054130                          .WORD   054130  ;TASK DESCRIPTOR
8581  034676  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
8582  034700  000240                          NOP                     ;RETURN HERE IF ERROR
8583  034702  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8584  034704  000137  035132                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8585  034710                          200$:
8586
8587  034710                          240$:
8588
8589                                  ;READ DATA FROM DEVICE
8590  034710  012737  107530  001402          MOV     #BUFTWO+4,RMBAO         ;CHANGE BUFFER
8591  034716  012737  000071  001376          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
8592  034724  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8593  034730  112722  000006                  MOVB    #RMDA,(R2)+
8594  034734  112722  000032                  MOVB    #RMOF,(R2)+
8595  034740  112722  000034                  MOVB    #RMDC,(R2)+
8596  034744  112722  000004                  MOVB    #RMBA,(R2)+
8597  034750  112722  000002                  MOVB    #RMWC,(R2)+
8598  034754  112722  000000                  MOVB    #RMCS1,(R2)+
8599  034760  112712  000200                  MOVB    #200,(R2)
8600
```

```
 8601  034764  004737  043246                 JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
 8602  034770  000404                         BR      250$                    ;GO TO 250$ IF NO ERROR
 8603  034772  000240                         NOP                     ;RETURN HERE IF ERROR
 8604  034774  104000                         ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
 8605  034776  000137  035132                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8606  035002                         250$:
 8607
 8608                                 ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
 8609  035002  004737  042712                 JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
 8610
 8611                                 ;WAIT FOR READ OPERATION TO COMPLETE
 8612  035006  004737  043606                 JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
 8613
 8614                                 ;GO READ STATUS FOR READ OPERATION
 8615  035012  004737  042776                 JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 8616  035016  000404                         BR      260$                    ;GO TO 260$ IF NO ERROR
 8617  035020  000240                         NOP                     ;RETURN HERE IF ERROR
 8618  035022  104000                         ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 8619  035024  000137  035132                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8620  035030                         260$:
 8621
 8622                                 ;CHECK FOR ERRORS DURING READ OPERATION
 8623  035030  004737  043772                 JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 8624  035034  000405                         BR      270$                    ;GO TO 270$ IF NO ERROR
 8625  035036  000240                         NOP                     ;RETURN HERE IF ERROR
 8626  035040  104000                         ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 8627  035042  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8628  035044  000137  035132                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8629  035050                         270$:
 8630  035050  004737  056276                 JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 8631  035054  000405                         BR      280$                    ;GO TO 280$ IF NO ERROR
 8632  035056  000240                         NOP                     ;RETURN HERE IF ERROR
 8633  035060  104000                         ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 8634  035062  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8635  035064  000137  035132                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8636  035070                         280$:
 8637
 8638  035070                         290$:
 8639  035070  004737  044624                 JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 8640  035074  000405                         BR      300$                    ;GO TO 300$ IF NO ERROR
 8641  035076  000240                         NOP                     ;RETURN HERE IF ERROR
 8642  035100  104000                         ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 8643  035102  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8644  035104  000137  035132                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8645  035110                         300$:
 8646  035110  004737  042344                 JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
 8647  035114  106524                         .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
 8648  035116  107530                         .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
 8649  035120  000404                         BR      310$                    ;GO TO 310$ IF NO ERROR
 8650  035122  000240                         NOP                     ;RETURN HERE IF ERROR
 8651  035124  104000                         ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
 8652  035126  000137  035132                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8653  035132                         310$:
 8654
 8655  035132                         350$:
 8656                                 ;;*****************************************************************
```

```
                                ;*TEST 27       WRITE, READ EACH TRACK
                                ;;*****************************************************************
035132                          TST27:
035132  000004                          SCOPE                   ;SCOPE CALL
035134  000240                          NOP                     ;START OF TEST
035136  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
035142  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
035146  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
035152  012737  000027  001226          MOV     #27,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                                ;;*****************************************************************
                                ;LOOP #1        FORMAT,WRITE,READ

035160                          10$:

                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
035160  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
035164  054130                  .WORD   054130  ;TASK DESCRIPTOR
035166  000404                  BR      20$                     ;GO TO 20$ IF NO ERROR
035170  000240                  NOP                     ;RETURN HERE IF ERROR
035172  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
035174  000137  036176          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
035200                          20$:

                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
035200  012737  000000  001432          MOV     #0,RMDC0                ;CYLINDER = 0
035206  012737  000000  001404          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
035214  012737  106520  001402  25$:     MOV     #BUFONE,RMBA0           ;BUS ADDRESS
035222  012737  177376  001400          MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
035230  012737  010000  001430          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
035236  012737  000063  001376          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND

                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE

035244  004737  040174                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
035250  000405                  BR      26$                     ;GO TO 26$ IF NO ERROR
035252  104401  066704          TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
035256  104000                  ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
035260  000137  036176          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
035264                          26$:
035264  012737  067756  001174          MOV     #MIXED,$TMP0            ;STARTING ADDRESS OF PATTERN
035272  012737  000200  001176          MOV     #128.,$TMP1             ;RANGE OF PATTERN
035300  004737  042100                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT

                                ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
035304  012702  001533                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
035310  112722  000034                  MOVB    #RMDC,(R2)+
035314  112722  000006                  MOVB    #RMDA,(R2)+
035320  112722  000004                  MOVB    #RMBA,(R2)+
035324  112722  000002                  MOVB    #RMWC,(R2)+
035330  112722  000032                  MOVB    #RMOF,(R2)+
035334  112722  000000                  MOVB    #RMCS1,(R2)+
035340  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
035344                          30$:

                                ;FORMAT THE DRIVE
```

```
 8713  035344  004737  043246                 JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
 8714  035350  000404                         BR      40$                     ;GO TO 40$ IF NO ERROR
 8715  035352  000240                         NOP                     ;RETURN HERE IF ERROR
 8716  035354  104000                         ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
 8717  035356  000137  036176                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8718  035362                         40$:
 8719
 8720                                 ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
 8721  035362  004737  042712                 JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
 8722
 8723                                 ;WAIT FOR THE FORMAT TO COMPLETE
 8724  035366  004737  043606                 JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
 8725
 8726                                 ;GO READ STATUS FOR FORMAT OPERATION
 8727  035372  004737  042776                 JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 8728  035376  000404                         BR      50$                     ;GO TO 50$ IF NO ERROR
 8729  035400  000240                         NOP                     ;RETURN HERE IF ERROR
 8730  035402  104000                         ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 8731  035404  000137  036176                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8732  035410                         50$:
 8733
 8734                                 ;VERIFY NO ERRORS DURING FORMAT
 8735  035410  004737  056276                 JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 8736  035414  000405                         BR      60$                     ;GO TO 60$ IF NO ERROR
 8737  035416  000240                         NOP                     ;RETURN HERE IF ERROR
 8738  035420  104000                         ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 8739  035422  004736                         JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8740  035424  000137  036176                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8741  035430                         60$:
 8742
 8743                                 ;MOVE LOOP ADDRESSES TO NEXT OPERATION
 8744  035430  012737  035446  001122             MOV     #70$,$LPADR
 8745  035436  012737  035446  001124             MOV     #70$,$LPERR
 8746  035444  000410                         BR      80$                             ;SKIP TO WRITE OPERATION
 8747
 8748                                 ;;*****************************************************************
 8749                                 ;LOOP #2        WRITE,READ
 8750
 8751  035446                         70$:
 8752
 8753
 8754                                 ;PREPARE DEVICE FOR WRITE OPERATION
 8755  035446  004737  037260                 JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 8756  035452  054130                         .WORD   054130  ;TASK DESCRIPTOR
 8757  035454  000404                         BR      80$                     ;GO TO 80$ IF NO ERROR
 8758  035456  000240                         NOP                     ;RETURN HERE IF ERROR
 8759  035460  104000                         ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 8760  035462  000137  036176                 JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8761  035466                         80$:
 8762
 8763
 8764  035466                         120$:
 8765
 8766                                 ;WRITE DATA TO THE DRIVE
 8767  035466  012737  106524  001402             MOV     #BUFONE+4,RMBAO         ;CHANGE BUS ADDRESS
 8768  035474  012737  177400  001400             MOV     #<-C256.+1>,RMWCO       ;CHANGE WORD COUNT
```

```
 8769  035502  012737  000061  001376          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
 8770  035510  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
 8771  035514  112722  000006                  MOVB    #RMDA,(R2)+
 8772  035520  112722  000034                  MOVB    #RMDC,(R2)+
 8773  035524  112722  000032                  MOVB    #RMOF,(R2)+
 8774  035530  112722  000004                  MOVB    #RMBA,(R2)+
 8775  035534  112722  000002                  MOVB    #RMWC,(R2)+
 8776  035540  112722  000000                  MOVB    #RMCS1,(R2)+
 8777  035544  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
 8778
 8779  035550  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
 8780  035554  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
 8781  035556  000240                          NOP                     ;RETURN HERE IF ERROR
 8782  035560  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
 8783  035562  000137  036176                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8784  035566                          130$:
 8785
 8786                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
 8787  035566  004737  042712                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
 8788
 8789                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
 8790  035572  004737  043606                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
 8791
 8792                                  ;GO READ STATUS FOR WRITE COMMAND
 8793  035576  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
 8794  035602  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
 8795  035604  000240                          NOP                     ;RETURN HERE IF ERROR
 8796  035606  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
 8797  035610  000137  036176                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8798  035614                          140$:
 8799
 8800                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
 8801  035614  004737  043772                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 8802  035620  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
 8803  035622  000240                          NOP                     ;RETURN HERE IF ERROR
 8804  035624  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 8805  035626  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8806  035630  000137  036176                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8807  035634                          150$:
 8808  035634  004737  056276                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 8809  035640  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
 8810  035642  000240                          NOP                     ;RETURN HERE IF ERROR
 8811  035644  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 8812  035646  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8813  035650  000137  036176                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8814  035654                          160$:
 8815
 8816  035654                          170$:
 8817  035654  004737  044624                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 8818  035660  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
 8819  035662  000240                          NOP                     ;RETURN HERE IF ERROR
 8820  035664  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 8821  035666  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 8822  035670  000137  036176                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
 8823  035674                          180$:
 8824
```

```
8825
8826  035674  012737  035712  001122          ;CHANGE LOOP ADDRESSES
                                                MOV     #190$,$LPADR
8827  035702  012737  035712  001124          MOV     #190$,$LPERR
8828  035710  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
8829
8830                                          ;;*****************************************************************
8831                                          ;LOOP #3        READ
8832
8833  035712                                  190$:
8834
8835                                          ;PREPARE DEVICE FOR READ OPERATION
8836  035712  004737  037260                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8837  035716  054130                          .WORD   054130  ;TASK DESCRIPTOR
8838  035720  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
8839  035722  000240                          NOP                     ;RETURN HERE IF ERROR
8840  035724  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8841  035726  000137  036176                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8842  035732                                  200$:
8843
8844  035732                                  240$:
8845
8846                                          ;READ DATA FROM DEVICE
8847  035732  012737  107530  001402          MOV     #BUFTWO+4,RMBAO         ;CHANGE BUS ADDRESS
8848  035740  012737  000071  001376          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
8849  035746  012702  001533                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8850  035752  112722  000006                  MOVB    #RMDA,(R2)+
8851  035756  112722  000032                  MOVB    #RMOF,(R2)+
8852  035762  112722  000034                  MOVB    #RMDC,(R2)+
8853  035766  112722  000004                  MOVB    #RMBA,(R2)+
8854  035772  112722  000002                  MOVB    #RMWC,(R2)+
8855  035776  112722  000000                  MOVB    #RMCS1,(R2)+
8856  036002  112712  000200                  MOVB    #200,(R2)
8857
8858  036006  004737  043246                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8859  036012  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
8860  036014  000240                          NOP                     ;RETURN HERE IF ERROR
8861  036016  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8862  036020  000137  036176                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8863  036024                                  250$:
8864
8865                                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8866  036024  004737  042712                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8867
8868                                          ;WAIT FOR READ OPERATION TO COMPLETE
8869  036030  004737  043606                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8870
8871                                          ;GO READ STATUS FOR READ OPERATION
8872  036034  004737  042776                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8873  036040  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
8874  036042  000240                          NOP                     ;RETURN HERE IF ERROR
8875  036044  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8876  036046  000137  036176                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8877  036052                                  260$:
8878
8879                                          ;CHECK FOR ERRORS DURING READ OPERATION
8880  036052  004737  043772                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
```

```
8881  036056  000405                          BR      270$            ;GO TO 270$ IF NO ERROR
8882  036060  000240                          NOP                     ;RETURN HERE IF ERROR
8883  036062  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8884  036064  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8885  036066  000137  036176                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
8886  036072                          270$:
8887  036072  004737  056276                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8888  036076  000405                          BR      280$            ;GO TO 280$ IF NO ERROR
8889  036100  000240                          NOP                     ;RETURN HERE IF ERROR
8890  036102  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8891  036104  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8892  036106  000137  036176                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
8893  036112                          280$:
8894
8895  036112                          290$:
8896  036112  004737  044624                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8897  036116  000405                          BR      300$            ;GO TO 300$ IF NO ERROR
8898  036120  000240                          NOP                     ;RETURN HERE IF ERROR
8899  036122  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8900  036124  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8901  036126  000137  036176                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
8902  036132                          300$:
8903  036132  004737  042344                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
8904  036136  106524                          .WORD   BUFONE+4        ;STARTING ADDRESS OF WRITE BUFFER
8905  036140  107530                          .WORD   BUFTWO+4        ;STARTING ADDDRESS OF READ BUFFER
8906  036142  000404                          BR      310$            ;GO TO 310$ IF NO ERROR
8907  036144  000240                          NOP                     ;RETURN HERE IF ERROR
8908  036146  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
8909  036150  000137  036176                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
8910  036154                          310$:
8911  036154  062737  000400  001404          ADD     #400,RMDA0      ;ADVANCE TO NEXT TRACK
8912  036162  022737  002037  001404          CMP     #2037,RMDA0     ;DONE??
8913  036170  103402                          BLO     350$            ;YES!!
8914  036172  000137  035214                  JMP     25$             ;TEST NEXT TRACK
8915
8916  036176                          350$:
8917  036176  012737  035160  001122          MOV     #10$,$LPADR     ;LOOP BACK TO START OF TEST
8918  036204  012737  035160  001124          MOV     #10$,$LPERR
8919                          ;PUT NEWTEST HERE
```

```
                                        $EOSP:
                                                NOP
                                                MOV     TSTQUE,R0
                                                ADD     #2,R0
                                                MOV     R0,TSTQUE
                                                TST     (R0)
                                                BEQ     1$
                                                JMP     READY
                                        1$:     MOV     #TSTQUE+2,TSTQUE
                                        .SBTTL  END OF PASS ROUTINE

                                        ;;**************************************************************
                                        ;;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
                                        ;;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
                                        ;;*IF THERES A MONITOR GO TO IT
                                        ;;*IF THERE ISN'T JUMP TO READY

                                        $EOP:
                                                NOP
                                                CLR     $TSTNM          ;;ZERO THE TEST NUMBER
                                                CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
                                                INC     $PASS           ;;INCREMENT THE PASS NUMBER
                                                BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
                                                DEC     (PC)+           ;;LOOP?
                                        $EOPCT: .WORD   1
                                                BGT     $DOAGN          ;;YES
                                                MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
                                        $ENDCT: .WORD   1
                                                $EOPCT
                                                TYPE    ,65$            ;;TYPE ASCIZ STRING
                                                BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <12><15>/END PASS #/
                                        64$:
                                                MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                                        ;;TYPE PASS NUMBER
                                                TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
                                                TYPE    ,67$            ;;TYPE ASCIZ STRING
                                                BR      66$             ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
                                        66$:
                                                MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
                                                                        ;;TOTAL NUMBER OF ERRORS
                                                TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
                                                TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
                                                CLR     $ERTTL          ;;CLEAR ERROR TOTAL
                                        $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
                                                BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
                                                RESET                   ;;CLEAR THE WORLD
                                        $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
                                                NOP                     ;;SAVE ROOM
                                                NOP                     ;;FOR
                                                NOP                     ;;ACT11
                                        $DOAGN:
                                                JMP     @(PC)+          ;;RETURN
                                        $RTNAD: .WORD   READY
```

```
 8976  036452     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
 8977           036456                         .EVEN
```

```
                                        .SBTTL  SUBROUTINES
                                        ;;*****************************************************************
                                        .SBTTL  ERROR TYPEOUT ROUTINE

                                        ;*THE ERROR TYPEOUT ROUTINE ASSEMBLES AND PRINTS INFORMATION
                                        ;*REGARDING THE DETECTION OF AN ERROR AS FOLLOWS:
                                        ;*
                                        ;*      .UNIT NUMBER, TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER ARE
                                        ;*PRINTED ON THE FIRST LINE;
                                        ;*      .ERROR MESSAGE IS ASSEMBLED, FORMATTED AND PRINTED ON
                                        ;*ONE OR MORE SUCCEEDING LINES;
                                        ;*      .PAIRED LINES OF ERROR HEADERS AND ERROR DATA
                                        ;*ARE PRINTED AFTER THE ERROR MESSAGE.

036456                                  ERRTYP:
036456  104414                                  SAVREG
036460  032777  020000  142466                  BIT     #SW13,@SWR      ;INHIBIT TYPEOUTS??
036466  001402                                  BEQ     1$              ;NO!!
036470  000137  037206                          JMP     21$             ;YES!!
                                        ;TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER
036474  104401  001217                 1$:     TYPE    ,$CRLF
036500  104401  037222                         TYPE    ,ERTY00         ;TYPE "UNT#"
036504  013746  001234                         MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
                                                                        ;;TYPE UNIT NUMBER
036510  104403                                 TYPOS                   ;;GO TYPE--OCTAL ASCII
036512     003                               .BYTE   3               ;;TYPE 3 DIGIT(S)
036513     000                               .BYTE   0               ;;SUPPRESS LEADING ZEROS
036514  005037  037212                         CLR     TSTNMB          ;LOAD TEST NUMBER FOR
036520  013737  001226  037212                MOV     $TESTN,TSTNMB
036526  104401  037230                         TYPE    ,ERTY01         ;TYPE "TST#"
036532  013746  037212                         MOV     TSTNMB,-(SP)    ;;SAVE TSTNMB FOR TYPEOUT
                                                                        ;;TYPE TEST NUMBER
036536  104403                                 TYPOS                   ;;GO TYPE--OCTAL ASCII
036540     003                               .BYTE   3               ;;TYPE 3 DIGIT(S)
036541     000                               .BYTE   0               ;;SUPPRESS LEADING ZEROS
036542  005037  037214                         CLR     ERRNMB          ;LOAD ERROR NUMBER FOR
036546  113737  001130  037214                MOVB    $ITEMB,ERRNMB   ;TYPEOUT
036554  001406                                  BEQ     2$              ;SKIP IF NO ERROR CALLED
036556  104401  037240                         TYPE    ,ERTY02         ;TYPE "ERR#"
036562  013746  037214                         MOV     ERRNMB,-(SP)    ;;SAVE ERRNMB FOR TYPEOUT
                                                                        ;;TYPE ERROR NUMBER
036566  104403                                 TYPOS                   ;;GO TYPE--OCTAL ASCII
036570     003                               .BYTE   3               ;;TYPE 3 DIGIT(S)
036571     000                               .BYTE   0               ;;SUPPRESS LEADING ZEROS
036572  104401  037247                 2$:     TYPE    ,ERTY03         ;TYPE "PC="
036576  013746  001132                         MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;TYPE PROGRAM COUNTER
036602  104403                                 TYPOS                   ;;GO TYPE--OCTAL ASCII
036604     006                               .BYTE   6               ;;TYPE 6 DIGIT(S)
036605     001                               .BYTE   1               ;;TYPE LEADING ZEROS
                                        ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
036606  005737  037214                 3$:     TST     ERRNMB          ;WAS AN ERROR CALLED?
036612  001575                                  BEQ     21$             ;NO!!
036614  104401  001217                         TYPE    ,$CRLF          ;YES-TYPE CRLF
036620  105037  037220                         CLRB    BOTFLG          ;CLEAR BOT FLAG
```

```
 9034  036624  105037  037221                CLRB    CHRCNT          ;CLEAR CHARACTER COUNTER
 9035  036630  013700  037214                MOV     ERRNMB,R0       ;R0 POINTS TO FIRST OF
 9036  036634  006300                        ASL     R0              ;FOUR ENTRIES IN ERROR
 9037  036636  006300                        ASL     R0              ;TABLE
 9038  036640  006300                        ASL     R0
 9039  036642  062700  001552                ADD     #$ERRTB-8.,R0
 9040  036646  011001                        MOV     (R0),R1         ;R1 POINTS TO ERROR MESSAGE
 9041                                                                ;TABLE
 9042  036650  001507                        BEQ     13$             ;BRANCH IF NO ERROR MESSAGE
 9043                                ;TYPE THE ERROR MESSAGE
 9044  036652  012102                4$:     MOV     (R1)+,R2        ;R2=ADDRESS OF MESSAGE STRING
 9045  036654  001505                        BEQ     12$             ;BRANCH IF END OF MESSAGE
 9046  036656  010237  037024                MOV     R2,11$          ;LOAD ADDRESS OF STRING
 9047  036662  005037  037216                CLR     BOTADR          ;CLEAR BOT ADDRESS
 9048  036666  112203                5$:     MOVB    (R2)+,R3        ;END OF STRING??
 9049  036670  001454                        BEQ     10$             ;YES!!
 9050  036672  122703  000015                CMPB    #CR,R3          ;CARRIAGE RETURN??
 9051  036676  001003                        BNE     6$              ;NO!!
 9052  036700  105037  037221                CLRB    CHRCNT          ;YES-CLEAR CHAR COUNT
 9053  036704  000770                        BR      5$              ;GET NEXT CHARACTER
 9054  036706  122703  000012                CMPB    #LF,R3          ;LINE FEED??
 9055  036712  001765                        BEQ     5$              ;YES-GET NEXT CHARACTER
 9056  036714  122703  000011                CMPB    #HT,R3          ;HORIZONTAL TAB??
 9057  036720  001007                        BNE     8$              ;NO!!
 9058  036722  105237  037221                INCB    CHRCNT          ;ADJUST CHARACTER COUNT
 9059  036726  132737  000007  037221        BITB    #7,CHRCNT
 9060  036734  001372                        BNE     7$
 9061  036736  000407                        BR      9$
 9062  036740  105237  037221          8$:   INCB    CHRCNT          ;INCREMENT CHARACTER COUNT
 9063  036744  122703  000040                CMPB    #' ,R3          ;SPACE??
 9064  036750  001002                        BNE     9$              ;NO!!
 9065  036752  010237  037216                MOV     R2,BOTADR       ;SAVE ADDRESS OF SPACE
 9066  036756  122737  000100  037221  9$:   CMPB    #64.,CHRCNT     ;END OF LINE??
 9067  036764  103340                        BHIS    5$              ;NO!!
 9068  036766  013704  037216                MOV     BOTADR,R4       ;GET ADDRESS OF LAST SPACE
 9069  036772  001007                        BNE     90$             ;BRANCH IF SPACE DETECTED
 9070  036774  104401  001217                TYPE    ,$CRLF          ;TYPE CRLF
 9071  037000  105037  037221                CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
 9072  037004  013702  037024                MOV     11$,R2          ;SET UP R2 FOR TESTING
 9073  037010  000726                        BR      5$
 9074  037012  105044                90$:    CLRB    -(R4)           ;REPLACE SPACE
 9075  037014  112737  177777  037220        MOVB    #-1,BOTFLG      ;SET BOT FLAG
 9076  037022  104401                10$:    TYPE                    ;TYPE ERROR MESSAGE STRING
 9077  037024  000000                11$:    .WORD                   ;STRING ADDRESS GOES HERE
 9078  037026  105737  037220                TSTB    BOTFLG          ;WAS STRING TRUNCATED??
 9079  037032  001707                        BEQ     4$              ;NO!!
 9080  037034  104401  001217                TYPE    ,$CRLF          ;YES-TYPE CRLF
 9081  037040  105037  037220                CLRB    BOTFLG          ;CLEAR BOT FLAG
 9082  037044  105037  037221                CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
 9083  037050  013702  037216                MOV     BOTADR,R2       ;SETUP R2 FOR TESTING
 9084  037054  010237  037024                MOV     R2,11$          ;SETUP 11$ FOR TYPING
 9085  037060  112742  000040                MOVB    #' ,-(R2)       ;RESTORE SPACE
 9086  037064  105722                        TSTB    (R2)+           ;RESTORE R2
 9087  037066  000677                        BR      5$              ;TYPE REST OF STRING
 9088  037070                          12$:
 9089                                ;TYPE ERROR HEADER AND ERROR DATA
```

```
                                13$:
037070                                  MOV     2(R0),R1        ;R1 POINTS TO ERROR HEADER TABLE
037070  016001  000002                   BEQ     21$             ;BRANCH IF NO HEADER
037074  001444                           TYPE    ,$CRLF          ;(ASSUME NO DATA)
037076  104401  001217                   MOV     4(R0),R2        ;R2 POINTS TO DATA ADDRESS TABLE
037102  016002  000004                   MOV     6(R0),R3        ;R3 POINTS TO FORMAT TABLE
037106  016003  000006        14$:       MOV     (R1)+,15$       ;PUT HEADER ADDRESS FOR TYPE
037112  012137  037122                   BEQ     21$             ;BRANCH IF END OF HEADERS
037116  001433                                                   ;(ASSUME END OF DATA)
                                        TYPE
037120  104401                15$:       .WORD   0               ;HEADER ADDRESS GOES HERE
037122  000000                          TYPE    ,$CRLF
037124  104401  001217                  TST     R2              ;DATA WITH HEADER??
037130  005702                          BEQ     14$             ;NO!!
037132  001767                          MOV     (R2)+,R4        ;R4 POINTS TO DATA ADDRESS
037134  012204                          MOV     (R3)+,R5        ;R5 POINTS TO FORMAT
037136  012305                16$:      TSTB    (R5)+           ;WHAT KIND OF DATA??
037140  105725                          BMI     18$             ;BINARY
037142  100407                          BEQ     17$             ;OCTAL
037144  001403                          MOV     @(R4)+,-(SP)    ;DECIMAL
037146  013446                          TYPDS
037150  104405                          BR      19$
037152  000405                17$:      MOV     @(R4)+,-(SP)
037154  013446                          TYPOC
037156  104402                          BR      19$
037160  000402                18$:      MOV     @(R4)+,-(SP)
037162  013446                          TYPBN
037164  104406                19$:      TST     (R4)            ;MORE DATA??
037166  005714                          BEQ     20$             ;NO!!
037170  001403                          TYPE    ,ERTY04         ;YES-TYPE 2 SPACES
037172  104401  037255                  BR      16$             ;AND CONTINUE
037176  000760                20$:      TYPE    ,$CRLF          ;TYPE ONE BLANK LINE
037200  104401  001217                  BR      14$             ;BEFORE NEXT HEADER
037204  000742                21$:      RESREG
037206  104415                          RTS     PC
037210  000207
                                TSTNMB: .WORD   0               ;TEST NUMBER
037212  000000                  ERRNMB: .WORD   0               ;ERROR NUMBER
037214  000000                  BOTADR: .WORD   0               ;BEGINNING OF TEXT ADDRESS
037216  000000                  BOTFLG: .BYTE   0               ;BOT FLAG
037220     000                  CHRCNT: .BYTE   0               ;CHARACTER COUNT
037221     000
037222  047125  052111  000043 ERTY00: .ASCIZ  @UNIT#@
037230  020054  042524  052123 ERTY01: .ASCIZ  @, TEST#@
037236  000043
037240  020054  051105  021522 ERTY02: .ASCIZ  @, ERR#@
037246     000
037247     054  050040  036503 ERTY03: .ASCIZ  @, PC=@
037254     000
037255     040  000040         ERTY04: .ASCIZ  @ @
                                .EVEN
```

```
                                .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

037260                          TSTPRP:

                                ;STORE TASK DESCRIPTOR AND CLEAR USER'S ERROR CALL
037260  017637  000000  040172          MOV     @(SP),500$      ;STORE DESCRIPTOR
037266  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
037272  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR CALL
037276  162716  000004                  SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
                                ;;*****************************************************************
                                ;SELECT DEVICE AND VERIFY DEVICE AVAILABLE IF BIT 15 SET IN TASK
037302  032737  100000  040172          BIT     #BIT15,500$     ;SELECT DEVICE??
037310  001414                          BEQ     30$             ;NO!!
037312  004737  050466                  JSR     PC,DEVSEL       ;GO SELECT DEVICE
037316  000411                          BR      30$             ;NO ERROR - CONTINUE
037320  000401                          BR      20$
037322  000000                  10$:    .WORD                   ;ERROR NUMBER FROM DEVSEL
037324  062716  000004          20$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
037330  113776  037322  000000          MOVB    10$,@(SP)
037336  000137  040160                  JMP     400$
037342                          30$:

                                ;;*****************************************************************
                                ;CLEAR CONTROLLER IF BIT 14 IS SET IN TASK
037342  032737  040000  040172          BIT     #BIT14,500$     ;CLEAR CONTROLLER??
037350  001453                          BEQ     120$            ;NO!!
```

```
 9198  037352  004737  052140                 JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 9199  037356  000411                         BR      60$             ;CONTINUE - NO ERROR
 9200  037360  000401                         BR      50$
 9201  037362  000000                 40$:    .WORD                   ;ERROR NUMBER FROM CNTCLR
 9202  037364  062716  000004         50$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9203  037370  113776  037362  000000         MOVB    40$,@(SP)
 9204  037376  000137  040160                 JMP     400$
 9205  037402                         60$:
 9206                                 ;;*****************************************************************
 9207                                 ;VERIFY CONTROLLER CLEAR IF BIT6 SET IN TASK
 9208  037402  032737  000100  040172         BIT     #BIT6,500$      ;VERIFY??
 9209  037410  001433                         BEQ     120$            ;NO!!
 9210  037412  004737  042712                 JSR     PC,GETSTS       ;SETUP INDEX TABLE
 9211  037416  004737  042776                 JSR     PC,GET          ;GO GET STATUS
 9212  037422  000411                         BR      90$             ;NO ERROR GETTING STATUS
 9213  037424  000401                         BR      80$
 9214  037426  000000                 70$:    .WORD                   ;ERROR FROM GETTING STATUS
 9215  037430  062716  000004         80$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9216  037434  113776  037426  000000         MOVB    70$,@(SP)
 9217  037442  000137  040160                 JMP     400$
 9218  037446  004737  052256         90$:    JSR     PC,CLRSTS       ;GO VERIFY STATUS CLEAR
 9219  037452  000412                         BR      120$            ;NO ERROR IN CLEAR
 9220  037454  000401                         BR      110$
 9221  037456  000000                 100$:   .WORD                   ;ERROR IN STATUS CLEAR
 9222  037460  005726                 110$:   TST     (SP)+           ;STRIP RETURN ADDRESS TO
 9223  037462  062716  000004                 ADD     #4,(SP)         ;SUBROUTINE AND TRANSFER
 9224  037466  113776  037456  000000         MOVB    100$,@(SP)      ;ERROR TO USER
 9225  037474  000137  040160                 JMP     400$
 9226  037500                         120$:
 9227
 9228                                 ;;*****************************************************************
 9229                                 ;EXECUTE PACK ACKNOWLEDGE IF BIT12 SET IN TASK AND VOLUME IS
 9230                                 ;NOT VALID
 9231  037500  032737  010000  040172         BIT     #BIT12,500$     ;PACK ACKNOWLEDGE??
 9232  037506  001511                         BEQ     240$            ;NO!!
 9233  037510  112737  000012  001504         MOVB    #RMDS,GETINX    ;GET RMDS
 9234  037516  112737  000200  001505         MOVB    #200,GETINX+1
 9235  037524  004737  042776                 JSR     PC,GET
 9236  037530  000411                         BR      150$            ;NO ERROR GETTING RMDS
 9237  037532  000401                         BR      140$
 9238  037534  000000                 130$:   .WORD
 9239  037536  062716  000004         140$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9240  037542  113776  037534  000000         MOVB    130$,@(SP)
 9241  037550  000137  040160                 JMP     400$
 9242  037554  032737  000100  001340 150$:   BIT     #VV,RMDSI       ;IS VOLUME VALID??
 9243  037562  001063                         BNE     240$            ;YES!!
 9244  037564  005037  001472                 CLR     MEDENB          ;CLEAR MEDIA ENABLE
 9245  037570  012737  000023  001376         MOV     #PAKACK!GO,RMCS1@       ;LOAD PACK ACK COMMAND
 9246  037576  112737  000000  001533         MOVB    #RMCS1,PUTINX   ;SETUP REGISTER INDEX TABLE
 9247  037604  112737  000200  001534         MOVB    #200,PUTINX+1
 9248  037612  004737  043246                 JSR     PC,PUT          ;GO WRITE COMMAND
 9249  037616  000410                         BR      180$            ;NO ERROR LOADING REGISTER
 9250  037620  000401                         BR      170$
 9251  037622  000000                 160$:   .WORD                   ;ERROR FROM PUT SUB
 9252  037624  062716  000004         170$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9253  037630  113776  037622  000000         MOVB    160$,@(SP)
```

```
9254  037636  000550                          BR      400$
9255  037640                          180$:
9256
9257                                  ;;*****************************************************************
9258                                  ;VERIFY PACK ACKNOWLEDGE IF #BIT4 SET IN TASK
9259  037640  032737  000020  040172          BIT     #BIT4,500$      ;VERIFY PACK ACKNOWLEDGE??
9260  037646  001431                          BEQ     240$            ;NO!!
9261  037650  004737  042712                  JSR     PC,GETSTS       ;SETUP FOR STATUS
9262  037654  004737  042776                  JSR     PC,GET          ;GO GET STATUS
9263  037660  000410                          BR      210$            ;NO ERROR GETTING STATUS
9264  037662  000401                          BR      200$
9265  037664  000000                  190$:   .WORD                   ;ERROR FROM GET SUB
9266  037666  062716  000004          200$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9267  037672  113776  037664  000000          MOVB    190$,@(SP)
9268  037700  000527                          BR      400$
9269  037702  004737  053136          210$:   JSR     PC,ACKSTS       ;GO CHECK ACKNOWLEDGE
9270  037706  000411                          BR      240$            ;NO ERROR
9271  037710  000401                          BR      230$
9272  037712  000000                  220$:   .WORD                   ;PACK ACKNOWLEDGE ERROR
9273  037714  005726                  230$:   TST     (SP)+           ;STRIP RETURN TO SUB AND
9274  037716  062716  000004                  ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9275  037722  113776  037712  000000          MOVB    220$,@(SP)
9276  037730  000513                          BR      400$
9277  037732                          240$:
9278
9279                                  ;;*****************************************************************
9280                                  ;RECALIBRATE DRIVE IF BIT 11 SET IN TASK AND PIP IS ACTIVE
9281  037732  032737  004000  040172          BIT     #BIT11,500$     ;RECALIBRATE??
9282  037740  001513                          BEQ     410$            ;NO!!
9283  037742  112737  000012  001504          MOVB    #RMDS,GETINX    ;LOAD REGISTER INDEX TABLE
9284  037750  112737  000200  001505          MOVB    #200,GETINX+1
9285  037756  004737  042776                  JSR     PC,GET          ;GO GET RMDS
9286  037762  000410                          BR      270$            ;NO ERROR GETTING RMDS
9287  037764  000401                          BR      260$
9288  037766  000000                  250$:   .WORD                   ;ERROR FROM GET SUB
9289  037770  062716  000004          260$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9290  037774  113776  037766  000000          MOVB    250$,@(SP)
9291  040002  000466                          BR      400$
9292  040004  032737  020000  001340  270$:   BIT     #PIP,RMDSI      ;IS PIP ACTIVE??
9293  040012  001466                          BEQ     410$            ;NO!!
9294  040014  012737  000007  001376          MOV     #RECAL!GO,RMCS10;LOAD RECALIBRATE COMMAND
9295  040022  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;AND REGISTER INDEX
9296  040030  112737  000200  001534          MOVB    #200,PUTINX+1
9297  040036  004737  043246                  JSR     PC,PUT          ;GO ISSUE RECALIBRATE
9298  040042  000410                          BR      300$            ;NO ERROR
9299  040044  000401                          BR      290$
9300  040046  000000                  280$:   .WORD                   ;ERROR IN REGISTER TRANSFER
9301  040050  062716  000004          290$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
9302  040054  113776  040046  000000          MOVB    280$,@(SP)
9303  040062  000436                          BR      400$
9304  040064  004737  042712          300$:   JSR     PC,GETSTS       ;SETUP FOR STATUS
9305  040070  004737  043606                  JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
9306
9307                                  ;;*****************************************************************
9308                                  ;VERIFY RECALIBRATE IF BIT 3 SET IN TASK
9309  040074  032737  000010  040172          BIT     #BIT3,500$      ;VERIFY RECALIBRATE??
```

```
 9310  040102  001432                        BEQ     410$            ;NO!!
 9311  040104  004737  042776                JSR     PC,GET          ;GO GET STATUS
 9312  040110  000410                        BR      330$            ;NO ERROR GETTING STATUS
 9313  040112  000401                        BR      320$
 9314  040114  000000              310$:     .WORD                   ;ERROR FROM GET
 9315  040116  062716  000004      320$:     ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9316  040122  113776  040114  000000        MOVB    310$,@(SP)
 9317  040130  000413                        BR      400$
 9318  040132  004737  053732      330$:     JSR     PC,RCLSTS       ;GO CHECK RECALIBRATE
 9319  040136  000414                        BR      410$            ;NO ERROR DURING RECALIBRATE
 9320  040140  000401                        BR      350$
 9321  040142  000000              340$:     .WORD                   ;ERROR DURING RECALIBRATE
 9322  040144  005726              350$:     TST     (SP)+           ;STRIP RETURN TO SUB AND
 9323  040146  062716  000004              ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9324  040152  113776  040142  000000        MOVB    340$,@(SP)
 9325  040160  162716  000002      400$:     SUB     #2,(SP)         ;MOVE SP BACK BEFORE ERROR
 9326  040164  000240                        NOP
 9327  040166  000240                        NOP
 9328  040170  000207              410$:     RTS     PC              ;RETURN TO USER
 9329
 9330  040172  000000              500$:     .WORD                   ;TASK/VERIFY DESCRIPTOR
```

```
                                .SBTTL  BAD SECTOR MODULE

                                ;THE BAD SECTOR MODULE PERFORMS TWO MAJOR FUNCTIONS, I.E.,
                                ;       RECOVER THE BAD SECTOR FILE FROM THE LAST TRACK, AND
                                ;       APPROVE THE USAGE OF A SECTOR BASED ON INFORMATION IN
                                ;THE BAD SECTOR FILE OR ASSIGN A NEW SECTOR IF THE ONE ELECTED IS
                                ;NOT AVAILABLE FOR USE.
                                ;
                                ;INFORMATION REQUIRED BY THE MODULE INCLUDES
                                ;       .RMDCO, THE DESIRED CYLINDER,
                                ;       .RMDAO, THE TRACK AND SECTOR ADDRESS,
                                ;       .RMWCO, THE WORD COUNT,
                                ;       .RMCS1O, THE COMMAND.
                                ;
                                ;MODULE CALL IS AS FOLLOWS,
                                ;
                                ;       JSR     PC,BADSCT
                                ;       BR      ???             ;RETURN HERE IF NO ERROR
                                ;       TYPE    ,MESSAGE        ;RETURN HERE IF THE BAD SECTOR FILE
                                ;                               CANNOT BE RECOVERED
                                ;       ERROR                   THE MODULE DEFINES THE ERROR NUMBER
                                ;
                                ;THE MODULE IS INTENDED TO BE CALLED PRIOR TO CALLING THE BUFFER
                                ;GENERATOR SUBROUTINE, AND PRESERVES THE "PUT BUFFER" SO THAT THE
                                ;BUFFER NEED ONLY BE FILLED ONCE FOR THE EXECUTION OF A FORMAT
                                ;OPERATION.
                                ;
                                ;THE MODULE RETURNS TO THE CALLING TEST WITH THE APPROVED OR ASSIGNED
                                ;SECTOR IN THE PUT BUFFER AND ALSO IN LOCATIONS "ASNDA" AND "ASNDC"
                                ;SO THAT A REFERENCE IS AVAILABLE TO THE TEST OUTSIDE OF THE PUT BUFFER.

 040174                         BADSCT:

                                ;TEST "MEDIA ENABLE" TO DETERMINE WHETHER OR NOT THE BAD SECTOR FILES
                                ;HAVE BEEN RECOVERED.
 040174  005737  001472                 TST     MEDENB
 040200  001402                         BEQ     10$
 040202  000137  041420                 JMP     300$            ;BAD SECTOR FILE IS AVAILABLE

                                ;;*****************************************************************
                                ;RECOVER THE BAD SECTOR FILE FROM THE LAST TRACK
 040206                         10$:

                                ;SAVE THE USER'S PUT BUFFER
 040206  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 040210  005000                         CLR     R0
 040212  016060  001376  106520 20$:    MOV     PUTBUF(R0),BUFFER(R0)
 040220  062700  000002                 ADD     #2,R0           ;ADVANCE TO NEXT BUFFER POSITION
 040224  022700  000046                 CMP     #46,R0          ;END OF BUFFER
 040230  103370                        BHIS    20$             ;NO !!

                                ;SET RETRY COUNT AND LOAD PUT BUFFER AND REGISTER INDEX TABLE
 040232  012737  000003  042066         MOV     #3,500$         ;RETRY COUNT
 040240  012737  002000  001404         MOV     #002000,RMDAO   ;STARTING SECTOR ADDRESS
 040246  012737  001466  001432         MOV     #822.,RMDCO     ;DESIRED CYLINDER
```

```
 9387  040254  012737  177376  001400          MOV     #<↑C258.+1>,RMWCO       ;WORD COUNT
 9388  040262  012737  010000  001430          MOV     #FMT16,RMOFO    ;16 BIT FORMAT
 9389  040270  012737  104500  001402          MOV     #MFGFIL,RMBAO   ;BUFFER ADDRESS
 9390
 9391  040276  012700  001533                  MOV     #PUTINX,R0      ;R0 POINTS TO REGISTER INDEX TABLE
 9392  040302  112720  000006                  MOVB    #RMDA,(R0)+
 9393  040306  112720  000034                  MOVB    #RMDC,(R0)+
 9394  040312  112720  000002                  MOVB    #RMWC,(R0)+
 9395  040316  112720  000032                  MOVB    #RMOF,(R0)+
 9396  040322  112720  000004                  MOVB    #RMBA,(R0)+
 9397  040326  112720  000000                  MOVB    #RMCS1,(R0)+
 9398  040332  112720  000200                  MOVB    #200,(R0)+
 9399  040336  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 9400
 9401                                  ;SET GET INDEX TABLE FOR READING STATUS
 9402  040340  004737  042712                  JSR     PC,GETSTS       ;SETUP GET INDEX REGISTER FOR STATUS
 9403  040344                          30$:
 9404
 9405                                  ;CLEAR THE DEVICE USING DRIVE CLEAR COMMAND
 9406  040344  012737  000011  001376          MOV     #DRVCLR!GO,RMCS10       ;LOAD COMMAND IN PUT BUFFER
 9407  040352  004737  043246                  JSR     PC,PUT          ;OUTPUT COMMAND
 9408  040356  000411                          BR      45$             ;RETURN HERE IF NO ERROR
 9409  040360  000401                          BR      40$             ;GET AROUND ERROR #
 9410  040362  000000                  35$:    .WORD                   ;ERROR # GOES HERE
 9411
 9412  040364  062716  000006          40$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
 9413  040370  113776  040362  000000          MOVB    35$,@(SP)       ;MOVE ERROR NUMBER TO USER
 9414  040376  000137  041076                  JMP     215$
 9415  040402  004737  042776          45$:    JSR     PC,GET          ;GO GET STATUS
 9416  040406  000411                          BR      60$             ;RETURN HERE IF NO ERROR
 9417  040410  000401                          BR      55$             ;GET AROUND ERROR #
 9418  040412  000000                  50$:    .WORD                   ;ERROR # GOES HERE
 9419
 9420  040414  062716  000006          55$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
 9421  040420  113776  040412  000000          MOVB    50$,@(SP)       ;MOVE ERROR # TO USERS ERROR CALL
 9422  040426  000137  041076                  JMP     215$
 9423
 9424  040432  004737  055474          60$:    JSR     PC,DRVSTS       ;GO VERIFY DRIVE CLEAR COMMAND
 9425  040436  000412                          BR      75$             ;RETURN HERE IF NO ERROR
 9426  040440  000401                          BR      70$             ;GET AROUND ERROR
 9427  040442  000000                  65$:    .WORD                   ;ERROR # GOES HERE
 9428
 9429  040444  005726                  70$:    TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
 9430  040446  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
 9431  040452  113776  040442  000000          MOVB    65$,@(SP)       ;MOVE ERROR # TO USER CALL
 9432  040460  000137  041076                  JMP     215$
 9433
 9434  040464                          75$:
 9435
 9436                                  ;ISSUE A PACK ACKNOWLEDGE IF VOLUME VALID IS RESET
 9437  040464  032737  000100  001340          BIT     #VV,RMDSI       ;IS VV RESET ??
 9438  040472  001050                          BNE     120$            ;NO !!
 9439  040474  012737  000023  001376          MOV     #PACACK!GO,RMCS10       ;LOAD COMMAND
 9440  040502  004737  043246                  JSR     PC,PUT          ;GO PUT COMMAND TO DRIVE
 9441  040506  000411                          BR      90$             ;RETURN HERE IF NO OUTPUT ERROR
 9442  040510  000401                          BR      85$             ;GET AROUND ERROR #
```

```
9443  040512  000000                  80$:    .WORD                   ;ERROR # GOES HERE
9444
9445  040514  062716  000006          85$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9446  040520  113776  040512  000000          MOVB    80$,@(SP)       ;MOVE ERROR # TO ERROR CALL
9447  040526  000137  041076                  JMP     215$
9448  040532  004737  042776          90$:    JSR     PC,GET          ;GO GET STATUS FOR PACK ACK
9449  040536  000411                          BR      105$            ;RETURN HERE IF NO ERROR
9450  040540  000401                          BR      100$            ;GET AROUND ERROR #
9451  040542  000000                  95$:    .WORD                   ;ERROR # GOES HERE
9452
9453  040544  062716  000006          100$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9454  040550  113776  040542  000000          MOVB    95$,@(SP)       ;MOVE ERROR # TO CALL
9455  040556  000137  041076                  JMP     215$
9456
9457  040562  004737  053136          105$:   JSR     PC,ACKSTS       ;GO VERIFY ACKNOWLEDGE STATUS
9458  040566  000412                          BR      120$            ;RETURN HERE IF NO ERROR
9459  040570  000401                          BR      115$            ;GET AROUND ERROR #
9460  040572  000000                  110$:   .WORD                   ;ERROR # GOES HERE
9461
9462  040574  005726                  115$:   TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
9463  040576  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9464  040602  113776  040572  000000          MOVB    110$,@(SP)      ;MOVE ERROR # TO USERS ERROR CALL
9465  040610  000137  041076                  JMP     215$
9466
9467  040614                          120$:
9468
9469                                  ;RECALIBRATE THE DRIVE IF PIP IS SET
9470  040614  032737  020000  001340          BIT     #PIP,RMDSI      ;IS PIP SET ??
9471  040622  001452                          BEQ     165$            ;NO !!
9472
9473  040624  012737  000007  001376          MOV     #RECAL!GO,RMCS10        ;LOAD RECALIBRATE COMMAND
9474  040632  004737  043246                  JSR     PC,PUT          ;PUT THE RECAL COMMAND
9475  040636  000411                          BR      135$            ;RETURN HERE IF NO ERROR
9476  040640  000401                          BR      130$            ;GET AROUND ERROR #
9477  040642  000000                  125$:   .WORD                   ;ERROR # GOES HERE
9478
9479  040644  062716  000006          130$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9480  040650  113776  040642  000000          MOVB    125$,@(SP)      ;MOVE ERROR # TO USERS CALL
9481  040656  000137  041076                  JMP     215$
9482  040662                          135$:
9483  040662  004737  043606                  JSR     PC,TIMOUT       ;WAIT FOR RECALIBRATE TO COMPLETE
9484  040666  004737  042776                  JSR     PC,GET          ;GO GET RECAL STATUS
9485  040672  000411                          BR      150$            ;RETURN HERE IF NO ERROR
9486  040674  000401                          BR      145$            ;GET AROUND ERROR #
9487  040676  000000                  140$:   .WORD                   ;ERROR # GOES HERE
9488
9489  040700  062716  000006          145$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9490  040704  113776  040676  000000          MOVB    140$,@(SP)      ;MOVE ERROR TO USERS CALL
9491  040712  000137  041076                  JMP     215$
9492
9493  040716  004737  053732          150$:   JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE STATUS
9494  040722  000412                          BR      165$            ;RETURN HERE IF NO ERROR
9495  040724  000401                          BR      160$            ;GET AROUND ERROR #
9496  040726  000000                  155$:   .WORD                   ;ERROR # GOES HERE
9497
9498  040730  005726                  160$:   TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
```

```
 9499  040732  062716  000006                 ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
 9500  040736  113776  040726  000000         MOVB    155$,@(SP)       ;MOVE ERROR # TO USERS CALL
 9501  040744  000137  041076                 JMP     215$
 9502
 9503  040750                         165$:
 9504
 9505                                 ;READ THE SECTOR IDENTIFIED BY RMDAO, INCLUDING HEADER AND DATA
 9506
 9507  040750  012737  000073  001376         MOV     #RH!GO,RMCS1O    ;LOAD READ HEADER AND DATA COMMAND
 9508  040756  004737  043246                 JSR     PC,PUT           ;PUT COMMAND
 9509  040762  000411                         BR      180$             ;RETURN HERE IF NO ERROR
 9510  040764  000401                         BR      175$             ;GET AROUND ERROR #
 9511  040766  000000                 170$:   .WORD                    ;ERROR # GOES HERE
 9512
 9513  040770  062716  000006         175$:   ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
 9514  040774  113776  040766  000000         MOVB    170$,@(SP)       ;MOVE ERROR # TO USERS ERROR CALL
 9515  041002  000137  041076                 JMP     215$
 9516
 9517  041006  004737  043606         180$:   JSR     PC,TIMOUT        ;WAIT FOR READ OPERATION TO COMPLETE
 9518  041012  004737  042776                 JSR     PC,GET           ;GO GET STATUS FOR READ OPERATION
 9519  041016  000411                         BR      195$             ;RETURN HERE IF NO ERROR
 9520  041020  000401                         BR      190$             ;GET AROUND ERROR #
 9521  041022  000000                 185$:   .WORD                    ;ERROR # GOES HERE
 9522
 9523  041024  062716  000006         190$:   ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
 9524  041030  113776  041022  000000         MOVB    185$,@(SP)       ;MOVE ERROR # TO CALL
 9525  041036  000137  041076                 JMP     215$
 9526
 9527  041042  004737  056276         195$:   JSR     PC,DTASTS        ;GO VERIFY RESULTS OF READ OPERATION
 9528  041046  000412                         BR      210$             ;RETURN HERE IF NO ERROR
 9529  041050  000401                         BR      205$             ;GET AROUND ERROR #
 9530  041052  000000                 200$:   .WORD                    ;ERROR # GOES HERE
 9531
 9532  041054  005726                 205$:   TST     (SP)+            ;STRIP RETURN ADDRESS TO SUBROUTINE
 9533  041056  062716  000006                 ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
 9534  041062  113776  041052  000000         MOVB    200$,@(SP)       ;MOVE ERROR # TO USERS CALL
 9535  041070  000137  041076                 JMP     215$
 9536
 9537  041074  000456                 210$:   BR      240$
 9538
 9539  041076                         215$:
 9540
 9541                                 ;AN ERROR HAS BEEN DETECTED IN TRYING TO READ THE BAD SECTOR FILE.
 9542                                 ;THE SECTOR WILL BE RETRIED IF POSSIBLE.
 9543
 9544  041076  032737  010000  001340         BIT     #MOL,RMDSI       ;IS MEDIUM ON LINE ??
 9545  041104  001446                         BEQ     230$             ;YES !!
 9546
 9547  041106  005337  042066                 DEC     500$             ;DECREMENT RETRY COUNT
 9548  041112  100037                         BPL     225$             ;RETRY IF COUNT NOT NEG
 9549
 9550                                 ;THE RETRY COUNT HAS EXPIRED - SEE IF THE ERROR IS MEDIA RELATED
 9551  041114  013746  001342                 MOV     RMER1I,-(SP)     ;GET ER1
 9552  041120  042716  100720                 BIC     #DCK!HCRC!HCE!FER!ECH,(SP)
 9553  041124  032737  000010  001370         BIT     #DPE,RMER2I      ;WAS THER A DATA PARITY ERROR ??
 9554  041132  001402                         BEQ     220$             ;NO !!
```

```
9555  041134  042716  000010            220$:   BIC     #PAR,(SP)       ;YES - CLEAR PAR
9556  041140  005726                            TST     (SP)+           ;ARE THERE ANY ERRORS NOT DUE TO MEDIA ?
9557  041142  001027                            BNE     230$            ;YES !!
9558  041144  013746  001370                    MOV     RMER2I,-(SP)    ;GET ER2
9559  041150  042726  100010                    BIC     #BSE!DPE,(SP)+  ;CLEAR MEDIA ERRORS
9560  041154  001022                            BNE     230$            ;BRANCH IF NONMEDIA ERRORS DETECTED
9561
9562                                    ;THE ERRORS DETECTED WHILE TRYING TO RECOVER THE BAD SECTOR FILE ARE
9563                                    ;DUE TO THE MEDIA.  SEE IF THE BAD SECTOR FILE CAN BE RECOVERED FROM
9564                                    ;ANOTHER AREA ON THE LAST TRACK
9565
9566  041156  062737  000002  001404            ADD     #2,RMDAO        ;ADVANCE SECTOR ADDRESS
9567  041164  122737  000012  001404            CMPB    #10.,RMDAO      ;QUIT IF ALL MFG SECTORS HAVE BEEN
9568  041172  001413                            BEQ     230$            ;TRIED
9569  041174  122737  000040  001404            CMPB    #32.,RMDAO      ;QUIT IF ALL USER SECTORS HAVE
9570  041202  001407                            BEQ     230$            ;BEEN TRIED
9571  041204  012737  000003  042066            MOV     #3,500$         ;REINSTATE RETRY COUNT FOR THIS SECTOR
9572  041212  162716  000006            225$:   SUB     #6,(SP)         ;MOVE SP BACK TO NO ERROR
9573  041216  000137  040344                    JMP     30$             ;RETRY THE READ OPERATION
9574
9575  041222                            230$:
9576
9577                                    ;THE BAD SECTOR FILE CANNOT BE READ
9578  041222  162716  000004                    SUB     #4,(SP)         ;MOVE SP TO ERROR RETURN
9579  041226  000137  042062                    JMP     410$            ;GO TO MODULE EXIT
9580
9581  041232                            240$:
9582
9583                                    ;THE SECTOR WAS RECOVERED WITHOUT ERROR -
9584                                    ;       READ THE USER FILE IF THIS IS THE MFG FILE
9585                                    ;       OR ELSE DONE
9586
9587  041232  122737  000011  001404            CMPB    #9.,RMDAO       ;WAS THE USER FILE READ ??
9588  041240  103451                            BLO     260$            ;YES - READ IS COMPLETE
9589
9590                                    ;IF 144 IS IMPLEMENTED THEN READ THE USER FILE -
9591                                    ;       ELSE DUMMY THE BAD SECTOR FILE
9592  041242  032737  140000  104500            BIT     #MSE!USE,MFGFIL ;ARE THE BAD SECTOR FLAGS OFF ??
9593  041250  001410                            BEQ     250$            ;YES - 144 IS DISABLED
9594
9595  041252  012737  002012  001404            MOV     #002012,RMDAO   ;READ THE USER FILE
9596  041260  012737  000003  042066            MOV     #3,500$         ;RELOAD THE RETRY COUNT FOR THIS SECTOR
9597  041266  000137  040344                    JMP     30$             ;GO READ THE USER FILE
9598  041272                            250$:
9599
9600                                    ;DUMMY THE BAD SECTOR FILES
9601  041272  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
9602  041274  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
9603  041276  012701  000374                    MOV     #252.,R1        ;R1 = NUMBER OF ENTRIES IN FILES
9604  041302  012700  000014                    MOV     #14,R0          ;R0 = ADDRESS INDEX TO FILE  STORAGE
9605  041306  012760  177777  104500    255$:   MOV     #-1,MFGFIL(R0)  ;ENTER ALL ONES IN MFG FILE
9606  041314  012760  177777  105510            MOV     #-1,USRFIL(R0)  ;ENTER ALL ONES IN USER FILE
9607  041322  062700  000002                    ADD     #2,R0           ;ADVANCE ADDRESS
9608  041326  005301                            DEC     R1              ;DECREMENT COUNT
9609  041330  001366                            BNE     255$            ;CONTINUE IF NOT DONE
9610  041332  012701  000006                    MOV     #6.,R1          ;CLEAR ID, SERIAL NUMBER ETC
```

```
 9611  041336  005000                        CLR     R0
 9612  041340  005060  104500          256$: CLR     MFGFIL(R0)
 9613  041344  005060  105510                CLR     USRFIL(R0)
 9614  041350  062700  000002                ADD     #2,R0
 9615  041354  005301                        DEC     R1
 9616  041356  001370                        BNE     256$
 9617
 9618  041360  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 9619  041362  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 9620  041364                          260$:
 9621
 9622                                  ;SET MEDIA ENABLE AND RESTORE THE USERS PUT BUFFER
 9623  041364  012737  177777  001472        MOV     #-1,MEDENB
 9624
 9625  041372  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 9626  041374  005000                        CLR     R0              ;R0 IS REGISTER INDEX
 9627  041376  016060  106520  001376  265$: MOV     BUFFER(R0),PUTBUF(R0)
 9628  041404  062700  000002                ADD     #2,R0           ;ADVANCE R0
 9629  041410  022700  000046                CMP     #46,R0          ;DONE ??
 9630  041414  103370                        BHIS    265$
 9631
 9632  041416  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 9633  041420                          270$:
 9634
 9635  041420                          300$:
 9636
 9637                                  ;;*****************************************************************
 9638                                  ;VERIFY THAT THE DESIRED SECTOR IS NOT IN THE MFG BAD SECTOR FILE
 9639                                  ;AND NOT IN THE USERS BAD SECTOR FILE.  ASSIGN A NEW SECTOR IF THE
 9640                                  ;SECTOR IS IN EITHER FILE.
 9641
 9642                                  ;LOAD INITIAL VARIABLES AND COMPUTE THE NUMBER OF SECTORS
 9643  041420  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 9644  041422  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 9645  041424  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 9646  041426  013737  001432  001474        MOV     RMDC0,ASNDC     ;LOAD REQUESTED CYLINDER, TRACK,
 9647  041434  013737  001404  001476        MOV     RMDA0,ASNDA     ;AND SECTOR ADDRESS IN ASSIGNED STORAGE
 9648  041442  005002                        CLR     R2              ;R2 = NUMBER OF SECTORS
 9649  041444  013700  001400                MOV     RMWC0,R0        ;R0 = WORD COUNT
 9650  041450  005300                        DEC     R0              ;CONVERT FROM 2'S COMPLEMENT
 9651  041452  005100                        COM     R0
 9652  041454  012701  000400                MOV     #256.,R1        ;R1 = NUMBER OF WORDS PER SECTOR
 9653  041460  032737  000002  001376        BIT     #BIT1,RMCS10    ;IS THIS A HEADER AND DATA COMMAND ??
 9654  041466  001402                        BEQ     305$            ;NO !!
 9655  041470  012701  000402                MOV     #258.,R1        ;CHANGE WORDS PER SECTOR
 9656  041474  020100                  305$: CMP     R1,R0           ;IS THERE A FULL SECTOR ??
 9657  041476  101404                        BLOS    310$            ;YES !!
 9658  041500  005700                        TST     R0              ;IS R0 ZERO ??
 9659  041502  001405                        BEQ     315$            ;YES !!
 9660  041504  005202                        INC     R2              ;INCREMENT FOR PARTIAL SECTOR
 9661  041506  000403                        BR      315$
 9662  041510  160100                  310$: SUB     R1,R0           ;SUBTRACT ONE SECTOR FROM WORD COUNT
 9663  041512  005202                        INC     R2              ;INCREMENT SECTOR COUNT
 9664  041514  000767                        BR      305$
 9665  041516  010237  042066          315$: MOV     R2,500$         ;SAVE SECTOR COUNT
 9666
```

```
 9667  041522                                316$:
 9668
 9669                                        ;LOAD PARAMETERS AND SEARCH THE MFG AND USER BAD SECTOR FILES FOR
 9670                                        ;THE ASSIGNED SECTOR AND ADJACENT SECTORS IF THE SECTOR COUNT IS
 9671                                        ;MORE THAN ONE.
 9672
 9673  041522  012737  104514  042076              MOV     #MFGFIL+14,540$ ;MOVE THE STARTING ADDRESS OF MFG
 9674                                                                      ;FILE TO BASE ADDRESS STORAGE
 9675  041530  013737  001474  042072  320$:       MOV     ASNDC,520$      ;LOAD COMPARING CYLINDER ADDRESS
 9676  041536  013737  001476  042074              MOV     ASNDA,530$      ;LOAD COMPARING TRACK, SECTOR ADDRESS
 9677  041544  013737  042066  042070              MOV     500$,510$       ;LOAD NUMBER OF SECTORS TO CONFIRM
 9678
 9679                                        ;SETUP FOR A BINARY SEARCH OF THE CURRENT FILE FOR THE COMPARING
 9680                                        ;CYLINDER, TRACK AND SECTOR ADDRESS
 9681  041552  013700  042076  325$:       MOV     540$,R0         ;LOAD THE BASE ADDRESS IN R0
 9682  041556  012701  000376              MOV     #<<127.*4>/2>,R1        ;R1 = LENGTH OF FILE
 9683  041562  060100                      ADD     R1,R0           ;START BINARY DIVIDE AT CENTER
 9684  041564  021037  042072  330$:       CMP     (R0),520$       ;DOES TABLE ENTRY = COMPARING CYLINDER ?
 9685  041570  001405                      BEQ     345$            ;YES !!
 9686  041572  101002                      BHI     340$            ;BRANCH IF ENTRY > COMPARING CYLINDER
 9687  041574                        335$:
 9688
 9689                                        ;FILE ENTRY IS LESS THAN COMPARING CYLINDER (OR TRACK AND SECTOR),
 9690                                        ;SO ADD DISPLACEMENT TO CURRENT POSITION
 9691  041574  060100                      ADD     R1,R0
 9692  041576  000410                      BR      350$
 9693  041600                        340$:
 9694
 9695                                        ;FILE ENTRY IS GREATER THAN COMPARING CYLINDER SO SUBTRACT DISPLACEMENT
 9696                                        ;FROM CURRENT POSITION
 9697  041600  160100                      SUB     R1,R0
 9698  041602  000406                      BR      350$
 9699  041604                        345$:
 9700
 9701                                        ;FILE ENTRY EQUALS COMPARING CYLINDER.  SEE IF THE NEXT ENTRY EQUALS
 9702                                        ;THE COMPARING TRACK, AND SECTOR.
 9703  041604  026037  000002  042074      CMP     2(R0),530$      ;ARE THEY EQUAL ??
 9704  041612  001413                      BEQ     360$            ;YES !!
 9705  041614  101371                      BHI     340$            ;BRANCH IF HIGHER
 9706  041616  000766                      BR      335$            ;BRANCH IF LOWER
 9707  041620                        350$:
 9708
 9709                                        ;THE POINTER (R0) HAS BEEN ADJUSTED.  HALVE THE DISPLACEMENT AND
 9710                                        ;CONTINUE THE SEARCH IF DISPLACEMENT NOT ZERO
 9711  041620  005701                      TST     R1              ;IS DISPLACEMENT ZERO ??
 9712  041622  001440                      BEQ     370$            ;YES !!
 9713  041624  006201                      ASR     R1              ;HALVE THE DISLPACEMENT
 9714  041626  042701  000003              BIC     #3,R1
 9715  041632  020037  042076              CMP     R0,540$         ;IS THE NEW POINTER WITHIN BOUNDS ??
 9716  041636  103432                      BLO     370$            ;NO !!
 9717  041640  000751                      BR      330$            ;CONTINUE SEARCH
 9718  041642                        360$:
 9719
 9720                                        ;THE COMPARING CYLINDER, TRACK AND SECTOR IS IN THE BAD SECTOR FILE.
 9721                                        ;ADVANCE THE ASSIGNED SECTOR AND START THE SEARCH ALL OVER.
 9722  041642  105237  001476              INCB    ASNDA           ;INCREMENT SECTOR
```

```
 9723  041646  122737  000037  001476          CMPB    #31.,ASNDA      ;SECTOR OK ??
 9724  041654  103022                          BHIS    365$            ;YES !!
 9725  041656  105037  001476                  CLRB    ASNDA           ;CLEAR SECTOR AND ADVANCE TRACK
 9726  041662  105237  001477                  INCB    ASNDA+1
 9727  041666  122737  000004  001477          CMPB    #4,ASNDA+1      ;TRACK OK ??
 9728  041674  103012                          BHIS    365$            ;YES !!
 9729  041676  005037  001476                  CLR     ASNDA           ;CLEAR TRACK AND SECTOR
 9730  041702  005237  001474                  INC     ASNDC           ;INCREMENT CYLINDER
 9731  041706  022737  001466  001474          CMP     #822.,ASNDC     ;CYLINDER OK ??
 9732  041714  103002                          BHIS    365$            ;YES !!
 9733  041716  005037  001474                  CLR     ASNDC
 9734  041722  000677                  365$:   BR      316$            ;REPEAT SEARCH
 9735
 9736  041724                          370$:
 9737
 9738                                  ;THE COMPARING SECTOR IS NOT IN THE BAD SECTOR FILES.  DECREMENT THE
 9739                                  ;NUMBER OF SECTORS TO COMPARE AND SEARCH THE NEXT SECTOR IF THE NUMBER
 9740                                  ;IS NOT ZERO.
 9741
 9742  041724  005337  042070          DEC     510$            ;DECREMENT NUMBER OF SECTORS TO COMPARE
 9743  041730  001432                  BEQ     380$            ;DONE IF ZERO
 9744  041732  105237  042074          INCB    530$            ;INCREMENT THE COMPARING SECTOR
 9745  041736  122737  000037  042074  CMPB    #31.,530$       ;SECTOR OK ??
 9746  041744  103022                  BHIS    375$            ;YES !!
 9747  041746  105037  042074          CLRB    530$            ;CLEAR SECTOR
 9748  041752  105237  042075          INCB    530$+1          ;INCREMENT TRACK
 9749  041756  122737  000004  042075  CMPB    #4,530$+1       ;TRACK OK ??
 9750  041764  103012                  BHIS    375$            ;YES !!
 9751  041766  005037  042074          CLR     530$            ;CLEAR SECTOR TRACK
 9752  041772  005237  042072          INC     520$            ;INCREMENT CYLINDER
 9753  041776  022737  001466  042072  CMP     #822.,520$      ;CYLINDER OK ??
 9754  042004  103002                  BHIS    375$            ;YES !!
 9755  042006  005037  042072          CLR     520$            ;CLEAR CYLINDER
 9756  042012  000137  041552          375$:   JMP     325$            ;SEARCH NEXT SECTOR
 9757
 9758  042016                          380$:
 9759
 9760                                  ;THE ASSIGNED SECTOR (AND REQUIRED ADJACENT SECTORS) ARE NOT IN
 9761                                  ;THE BAD SECTOR FILE JUST SEARCHED.  SEARCH THE USER FILE IF IT
 9762                                  ;HAS NOT YET BEEN SEARCHED, ELSE ASSIGN THE SECTOR AND RETURN TO USER.
 9763
 9764  042016  022737  105524  042076          CMP     #USRFIL+14,540$ ;TEST BASE ADDRESS
 9765  042024  001405                          BEQ     400$            ;DONE IF BASE = USER
 9766  042026  012737  105524  042076          MOV     #USRFIL+14,540$ ;LOAD BASE ADDRESS
 9767  042034  000137  041530          390$:   JMP     320$            ;SEARCH USER FILE
 9768
 9769
 9770  042040                          400$:
 9771
 9772                                  ;ASSIGN THE SECTOR AND RETURN TO USER
 9773  042040  013737  001474  001432          MOV     ASNDC,RMDC0     ;LOAD CYLINDER
 9774  042046  013737  001476  001404          MOV     ASNDA,RMDA0     ;LOAD TRACK AND SECTOR
 9775  042054  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 9776  042056  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 9777  042060  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 9778  042062  000240                  410$:   NOP
```

```
 9779  042064  000207                    RTS     PC
 9780
 9781                                    ;THE FOLLOWING ARE STORAGE LOCATIONS FOR THE MODULE
 9782
 9783  042066  000000                    500$:   .WORD   ;RETRY COUNT/ NUMBER OF SECTORS REQUIRED
 9784  042070  000000                    510$:   .WORD   ;NUMBER OF SECTORS TO COMPARE
 9785  042072  000000                    520$:   .WORD   ;COMPARING CYLINDER
 9786  042074  000000                    530$:   .WORD   ;COMPARING TRACK AND SECTOR
 9787  042076  000000                    540$:   .WORD   ;BASE ADDRESS OF BAD SECTOR FILE BEING SEARCHED
```

```
                        .SBTTL  BUFFER GENERATOR SUBROUTINE

                        ; THIS SUBROUTINE GENERATES A DATA BUFFER FOR WRITE COMMANDS.  THE
                        ; BUFFER STARTS AT RMBA AND IS RMWC WORDS LONG.  THE BUFFER
                        ; CONTAINS A REPETITIVE DATA PATTERN CONSISTING OF $TMP1 WORDS
                        ; FROM THE DATA PATTERN TABLE, BEGINNING AT ADDRESS $TMP0.
                        ; HEADER INFORMATION FOR THE BUFFER IS EXTRACTED FROM RMDC,
                        ; RMDA AND RMOF.

                        ;CALL:
                        ;(1)    JSR     PC,GENBUF
                        ;(2)    ??                              RETURN HERE

042100                  GENBUF:
042100  010046                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
042102  010146                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
042104  010246                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
042106  010346                 MOV     R3,-(SP)        ;;PUSH R3 ON STACK
042110  010446                 MOV     R4,-(SP)        ;;PUSH R4 ON STACK
042112  013700  001402         MOV     RMBA0,R0        ;LOAD DATA BUFFER ADDRESS
042116  013701  001400         MOV     RMWC0,R1        ;LOAD WORD COUNT
042122  013737  001432 042340 MOV     RMDC0,60$       ;LOAD STARTING CYLINDER ADDRESS
042130  013737  001404 042342 MOV     RMDA0,65$       ;LOAD STARTING TRACK,SECTOR ADDRESS
042136  032737  000002 001376 10$:    BIT     #BIT1,RMCS1     ;WRITE HEADER & DATA??
042144  001450                 BEQ     25$             ;NO!!
042146  013710  042340         MOV     60$,(R0)        ;WRITE HEADER WORD #1
042152  042710  176000         BIC     #^C<CYLMSK>,(R0)
042156  052710  140000         BIS     #MSE!USE,(R0)   ;SET (DISABLE) BAD SECTOR FLAGS
042162  012702  000035         MOV     #29.,R2         ;R2 = MAXIMUM SECTOR ADDRESS
042166  032737  010000 001430 BIT     #FMT16,RMOF0    ;16 BIT FORMAT??
042174  001404                 BEQ     15$             ;NO!!
042176  052710  010000         BIS     #FMT16,(R0)     ;SET FORMAT BIT IN HEADER
042202  012702  000037         MOV     #31.,R2         ;CHANGE MAXIMUM SECTOR ADDRESS
042206  005201         15$:    INC     R1              ;INCREMENT WORD COUNT
042210  001444                 BEQ     50$             ;EXIT IF DONE
042212  062700  000002         ADD     #2,R0           ;MOVE R0 TO HEADER WORD #2
042216  013720  042342         MOV     65$,(R0)+       ;WRITE HEADER WORD #2
042222  005201                 INC     R1              ;INCREMENT WORD COUNT AND
042224  001436                 BEQ     50$             ;EXIT IF DONE
042226  012703  042342         MOV     #65$,R3         ;ADVANCE SECTOR ADDRESS
042232  105213                 INCB    (R3)
042234  120213                 CMPB    R2,(R3)         ;SECTOR OVERFLOW ??
042236  103013                 BHIS    25$             ;NO !!
042240  105013                 CLRB    (R3)            ;YES - CLEAR SECTOR ADDRESS
042242  105263  000001         INCB    1(R3)           ;ADVANCE TRACK ADDRESS
042246  122763  000004 000001 CMPB    #4,1(R3)        ;TRACK OVERFLOW ??
042254  103004                 BHIS    25$             ;NO !!
042256  105063  000001         CLRB    1(R3)           ;YES - CLEAR TRACK ADDRESS
042262  105237  042340         INCB    60$             ;ADVANCE CYLINDER ADDRESS
042266  012704  000400 25$:    MOV     #256.,R4                ;LOAD SECTOR DATA COUNT
042272  013702  001174 30$:    MOV     $TMP0,R2        ;LOAD PATTERN ADDRESS
042276  013703  001176         MOV     $TMP1,R3        ;LOAD PATTERN COUNT
042302  012220         40$:    MOV     (R2)+,(R0)+     ;WRITE DATA PATTERN
042304  005201                 INC     R1              ;INCREMENT WORD COUNT AND
042306  001405                 BEQ     50$             ;EXIT IF DONE
042310  005304                 DEC     R4              ;DECREMENT SECTOR COUNT
```

```
 9844  042312  001711                    BEQ     10$             ;START NEXT SECTOR IF 0
 9845  042314  005303                    DEC     R3              ;DECREMENT PATTERN COUNT
 9846  042316  001765                    BEQ     30$             ;RESTART PATTERN IF 0
 9847  042320  000770                    BR      40$             ;CONTINUE DATA PATTERN
 9848
 9849  042322                    50$:
 9850  042322  012604                    MOV     (SP)+,R4        ;;POP STACK INTO R4
 9851  042324  012603                    MOV     (SP)+,R3        ;;POP STACK INTO R3
 9852  042326  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
 9853  042330  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
 9854  042332  012600                    MOV     (SP)+,R0        ;;POP STACK INTO R0
 9855  042334  000240                    NOP
 9856  042336  000207                    RTS     PC
 9857
 9858  042340  000000            60$:    .WORD                   ;CYLINDER ADDRESS STORAGE
 9859  042342  000000            65$:    .WORD                   ;TRACK, SECTOR ADDRESS STORAGE
```

```
 9860                                   .SBTTL  COMPARE BUFFER SUBROUTINE
 9861
 9862                                   ;THIS SUBROUTINE COMPARES THE CONTENTS OF BUFONE AND BUFTWO,
 9863                                   ;ASSUMING THAT BUFONE IS THE BUFFER FROM WHICH DATA WAS WRITTEN
 9864                                   ;AND BUFTWO IS THE BUFFER TO WHICH DATA WAS READ.  ERRORS IN BUFFER
 9865                                   ; COMPARISON ARE REPORTED TO THE USER VIA THE USER'S ERROR CALL.
 9866
 9867                                   ;CALL:
 9868                                   ;(1)    JSR     PC,CMPBUF
 9869                                   ;(2)    .WORD   WRITE BUFFER ADDRESS
 9870                                   ;       .WORD   READ BUFFER ADDRESS
 9871                                   ;(3)    BR      ??              RETURN HERE IF NO ERROR
 9872                                   ;(4)    NOP                     RETURN HERE IF ERROR
 9873                                   ;(5)    ERROR                   ERROR DEFINED BY SUBROUTINE
 9874                                   ;(6)    ???
 9875
 9876  042344                           CMPBUF:
 9877  042344  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 9878  042346  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 9879  042350  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 9880  042352  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 9881
 9882  042354  005037  042710                  CLR     150$            ;CLEAR CORRECTION FLAG
 9883                                   ;DETERMINE IF DATA SHOULD BE CORRECTED
 9884  042360  032737  100000  001342          BIT     #DCK,RMER1I     ;WAS THERE A DATA CHECK ??
 9885  042366  001465                          BEQ     80$             ;NO !!
 9886  042370  032737  000100  001342          BIT     #ECH,RMER1I     ;IS IT A HARD ERROR ??
 9887  042376  001061                          BNE     80$             ;YES !!
 9888  042400  033737  004000  001360          BIT     ECI,RMOFI       ;WAS ECC CORRECTION ALLOWED ??
 9889  042406  001055                          BNE     80$             ;NO !!
 9890  042410  032737  010000  001360          BIT     #FMT16,RMOFI    ;IS THIS 16 BIT FORMAT ??
 9891  042416  001451                          BEQ     80$             ;NO !!
 9892
 9893                                   ;CORRECT DATA USING ECC INFORMATION
 9894  042420  013700  001402                  MOV     RMBAO,R0        ;R0 = STARTING BUFFER ADDRESS
 9895  042424  013701  001372                  MOV     RMEC1I,R1       ;R1 = ECC POSITION
 9896  042430  052737  100000  042710          BIS     #BIT15,150$     ;SET CORRECTION FLAG
 9897                                   ;MOVE R0 TO WORD BOUNDARY OF ERROR BURST
 9898
 9899  042436  022701  000020           10$:    CMP     #16.,R1         ;IS BIT POSITION > 1 WORD
 9900  042442  103005                          BHIS    20$             ;NO !!
 9901  042444  162701  000020                  SUB     #16.,R1         ;SUBTRACT 1 WORDS WORTH
 9902  042450  062700  000002                  ADD     #2,R0           ;ADVANCE BUFFER ADDRESS 1 WORD
 9903  042454  000770                          BR      10$
 9904  042456  012702  000001           20$:    MOV     #1,R2           ;R2 = BIT POINTER
 9905  042462  010203                          MOV     R2,R3           ;R3 = BIT NUMBER
 9906                                   ;MOVE R2 TO STARTING BIT OF ERROR BURST
 9907
 9908  042464  020301                   30$:    CMP     R3,R1           ;IS R3 SAME AS R1 ??
 9909  042466  001403                          BEQ     35$             ;YES !!
 9910  042470  006302                          ASL     R2              ;SHIFT BIT POINTER
 9911  042472  005203                          INC     R3              ;INCREMENT BIT NUMBER
 9912  042474  000773                          BR      30$
 9913  042476  012703  000013           35$:    MOV     #11.,R3         ;R3 = LENGTH OF ERROR BURST
 9914
 9915                                   ;CORRECT THE ERROR BURST
```

```
 9916  042502  030237  001374          40$:    BIT     R2,RMEC2I       ;IS THIS BIT SET IN ECC PATTERN ??
 9917  042506  001405                          BEQ     60$             ;NO - DO NOT CORRECT THIS BIT
 9918  042510  030210                          BIT     R2,(R0)         ;IS THE BIT PRESENTLY SET ??
 9919  042512  001402                          BEQ     50$             ;NO
 9920  042514  040210                          BIC     R2,(R0)         ;RESET THE BIT
 9921  042516  000401                          BR      60$
 9922  042520  050210                  50$:    BIS     R2,(R0)         ;SET THE BIT
 9923  042522  006302                  60$:    ASL     R2              ;SHIFT TO NEXT BIT
 9924  042524  001004                          BNE     70$
 9925  042526  012702  000001                  MOV     #1,R2           ;CONTINUE WITH FIRST BIT OF NEXT WORD
 9926  042532  062700  000002                  ADD     #2,R0
 9927  042536  005303                  70$:    DEC     R3              ;END OF BURST ??
 9928  042540  001360                          BNE     40$             ;NO !!
 9929  042542  017600  000010          80$:    MOV     @10(SP),R0              ;R0 = WRITE BUFFER
 9930  042546  062766  000002  000010          ADD     #2,10(SP)               ;MOVE SP TO READ ADDRESS
 9931  042554  017601  000010                  MOV     @10(SP),R1              ;R1 = READ BUFFER
 9932  042560  062766  000002  000010          ADD     #2,10(SP)               ;MOVE SP TO RETURN ADDRESS
 9933  042566  013702  001330                  MOV     RMWCI,R2                ;R2 = NUMBER OF WORDS TRANSFER
 9934  042572  163702  001400                  SUB     RMWCO,R2
 9935  042576  022021                  90$:    CMP     (R0)+,(R1)+             ;COMPARE DATA WORDS
 9936  042600  001003                          BNE     100$                    ;EXIT IF NOT EQUAL
 9937  042602  005302                          DEC     R2                      ;DECREMENT WORD COUNT
 9938  042604  001374                          BNE     90$                     ;CONTINUE IF NOT DONE
 9939  042606  000433                          BR      110$                    ;DONE COMPARE - NO ERROR
 9940  042610                          100$:
 9941  042610  014037  001140                  MOV     -(R0),$GDDAT            ;STORE GOOD DATA FOR TYPEOUT
 9942  042614  014137  001142                  MOV     -(R1),$BDDAT            ;STORE BAD DATA FOR TYPEOUT
 9943  042620  010037  001134                  MOV     R0,$GDADR               ;STORE ADDRESS OF GOOD DATA
 9944  042624  010137  001136                  MOV     R1,$BDADR               ;STORE ADDRESS OF BAD DATA
 9945  042630  010237  001174                  MOV     R2,$TMP0                ;STORE WORD COUNT OF ERROR
 9946  042634  062766  000004  000010          ADD     #4,10(SP)               ;MOVE SP TO USER'S ERROR CALL
 9947  042642  112776  000336  000010          MOVB    #336,@10(SP)            ;WRITE ERROR NUMBER IN CALL
 9948
 9949                          ;CHANGE ERROR NUMBER IF ECC CORRECTION FAILED
 9950  042650  032737  100000  042710          BIT     #BIT15,150$     ;WAS ECC CORRECTION USED ??
 9951  042656  001403                          BEQ     105$            ;NO !!
 9952  042660  112776  000163  000010          MOVB    #163,@10(SP)    ;ECC CORRECTION FAILED
 9953  042666  162766  000002  000010  105$:   SUB     #2,10(SP)               ;MOVE SP TO RETURN IF ERROR
 9954  042674  000240                          NOP
 9955  042676                          110$:
 9956  042676  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
 9957  042700  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 9958  042702  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 9959  042704  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 9960  042706  000207                          RTS     PC                      ;RETURN TO USER
 9961
 9962  042710  000000                  150$:   .WORD                   ;ECC CORRECTION FLAG
```

```
                                        .SBTTL  GET STATUS SUBROUTINE
                                        ;
                                        ;THIS SUBROUTINE SETS UP THE "GET INDEX TABLE" AND THE "GET
                                        ;BUFFER" FOR READING ALL SUBSYSTEM REGISTERS VIA THE GET SUBROUTINE
                                        ;AND THEN RETURNS TO THE USER.
                                        ;
                                        ;CALL:  JSR     PC,GETSTS
                                        ;       ???                     RETURN HERE
                                        ;
042712                                  GETSTS:
042712  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
042714  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
042716  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
042720  012700  001504                          MOV     #GETINX,R0      ;R0= ADDRESS OF INDEX TABLE
042724  012701  001376                          MOV     #RMEC2I+2,R1    ;R1 = ADDRESS OF GET BUFFER
042730  012702  000046                          MOV     #RMEC2,R2       ;R2 = REGISTER INDE
042734  110220                        2$:       MOVB    R2,(R0)+        ;WRITE REGISTER INDEX IN TABLE
042736  005041                                  CLR     -(R1)           ;CLEAR CORRESPONDING LOCATION
042740  162702  000002                3$:       SUB     #2,R2           ;DECREMENT TO NEXT INDEX
042744  100405                                  BMI     4$              ;BRANCH OUT IF DONE
042746  022702  000022                          CMP     #RMDB,R2        ;DONT WRITE RMDB INDEX
042752  001370                                  BNE     2$
042754  005041                                  CLR     -(R1)
042756  000770                                  BR      3$
042760  112720  000200                4$:       MOVB    #200,(R0)+      ;WRITE TERMINATOR
042764  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
042766  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
042770  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
042772  000240                                  NOP
042774  000207                                  RTS     PC
```

```
                        .SBTTL  GET SUBROUTINE

                        ;THIS SUBROUTINE READS THE REGISTERS WHICH ARE LISTED IN THE
                        ;"GET INDEX TABLE" AND STORES THEIR VALUES IN THE CORRESPONDING
                        ;LOCATION IN THE "GET REGISTER BUFFER".  FOR EXAMPLE, AN
                        ;ENTRY OF 04 IN THE TABLE WILL CAUSE THE SUBROUTINE TO
                        ;READ "RMBA" AND STORE ITS CONTENTS AT THE LOCATION IN
                        ;THE BUFFER ASSIGNED TO THAT REGISTER.  THE NUMBER OF
                        ;REGISTERS TO BE READ IS VARIABLE FROM 1 TO 22; THE INDEX
                        ;TABLE MUST BE TERMINATED WITH A CONTROL BYTE (200)
                        ;WHICH SHOULD FOLLOW THE LAST ENTRY.
                        ;
                        ;SUBROUTINE CALL:
                        ;(1)     "GET INDEX TABLE" HAS BEEN LOADED WITH REGISTER INDEX
                        ;       VALUES AND TERMINATED WITH A CONTROL BYTE
                        ;(2)     "GET INPUT BUFFER" IS AVAILABLE FOR USE.  (NOTE THAT
                        ;       UNUSED LOCATIONS, I.E.,ENTRIES IN BUFFER CORRESPONDING
                        ;       TO REGISTERS NOT READ, ARE NOT CHANGED.)
                        ;(3)     JSR     PC,GET
                        ;       BR      ???             RETURN HERE IF NO ERROR FOUND
                        ;       NOP                     RETURN HERE IF ANY ERROR FOUND
                        ;       ERROR                   SUB DEFINES ERROR NUMBER
                        ;       ???
                        ;
                        GET:    NOP
                                ADD     #4,(SP)         ;CLEAR ERROR NUMBER IN USER'S
                                CLRB    @(SP)           ;ERROR CALL
                                SUB     #4,(SP)
                                MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                                MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                                MOV     R2,-(SP)        ;;PUSH R2 ON STACK
                                MOV     R3,-(SP)        ;;PUSH R3 ON STACK
                                MOV     R4,-(SP)        ;;PUSH R4 ON STACK
                                MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
                                MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
                                MOV     $BASE,R0
                                MOV     #GETBUF,R2
                                MOV     #GETINX,R4
                                MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
                                MOV     #PR6,ERRVEC+2
                        1$:     MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
                                MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
                                BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
                                BNE     3$              ;YES!!
                                ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN USER'S
                                MOVB    #112,@16(SP)    ;ERROR CALL
                                BR      7$
                        3$:     TSTB    (R4)            ;DONE??
                                BMI     9$              ;YES!!
                                MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
                                BIC     #^CIDXMSK,R1    ;CLEAR ANY SIGN EXTENSION
                                ADD     R0,R1
                                MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS FOR REGISTER
                                BIC     #^CIDXMSK,R3    ;CLEAR ANY SIGN EXTENSION
                                ADD     R2,R3
                                MOV     (R1),(R3)       ;READ REGISTER
```

```
10050  043156  000764                        BR      3$
10051
10052  043160  022626                5$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
10053  043162  062766  000004  000016        ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
10054  043170  112776  000007  000016        MOVB    #7,@16(SP)      ;USER'S ERROR CALL
10055  043176  162766  000002  000016  7$:   SUB     #2,16(SP)
10056  043204  105714                8$:     TSTB    (R4)            ;DONE CLEARING??
10057  043206  100405                        BMI     9$              ;YES!!
10058  043210  005003                        CLR     R3              ;CLEAR REMAINING STORAGE
10059  043212  112403                        MOVB    (R4)+,R3        ;LOCATIONS
10060  043214  060203                        ADD     R2,R3
10061  043216  005013                        CLR     (R3)
10062  043220  000771                        BR      8$
10063  043222                        9$:
10064  043222  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10065  043226  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10066  043232  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
10067  043234  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
10068  043236  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
10069  043240  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
10070  043242  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
10071  043244  000207                        RTS     PC              ;RETURN
10072
```

```
                                .SBTTL  PUT SUBROUTINE

                                ;THIS SUBROUTINE WRITES THE REGISTERS WHICH ARE LISTED IN THE
                                ;"PUT INDEX TABLE" WITH  THE CONTENTS OF THE CORRESPONDING
                                ;LOCATION IN THE "PUT REGISTER BUFFER".  THE NUMBER OF
                                ;REGISTERS WRITTEN IS VARIABLE; THE INDEX TABLE MUST
                                ;BE TERMINATED WITH A CONTROL BYTE (200) WHICH SHOULD
                                ;FOLLOW THE LAST ENTRY.
                                ;
                                ;SUBROUTINE CALL:
                                ;
                                ;(1)     "PUT INDEX TABLE" HAS BEEN LOADED WITH INDEX VALUES
                                ;        OF REGISTERS TO BE WRITTEN.
                                ;(2)     "PUT REGISTER BUFFER" CONTAINS CONTENTS OF EACH
                                ;        REGISTER TO BE WRITTEN.
                                ;(3)     JSR     PC,PUT
                                ;        BR      ???             RETURN HERE IF NO ERROR FOUND
                                ;        NOP                     RETURN HERE IF ANY ERROR FOUND
                                ;        ERROR                   SUB DEFINES ERROR NUMBER
                                ;        ???
                                ;
043246  000240          PUT:    NOP
043250  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
043252  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
043254  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
043256  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
043260  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
043262  013746  000004          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
043266  013746  000006          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
043272  013700  001276          MOV     $BASE,R0
043276  012702  001376          MOV     #PUTBUF,R2
043302  012704  001533          MOV     #PUTINX,R4
043306  012737  043414  000004  MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
043314  012737  000300  000006  MOV     #PR6,ERRVEC+2
043322  016037  000010  001174  1$:     MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
043330  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
043336  032737  004000  001176          BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
043344  001007                  BNE     3$              ;YES!!
043346  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
043354  112776  000112  000016          MOVB    #112,@16(SP)    ;USER'S ERROR CALL
043362  000423                  BR      7$
043364  105714          3$:     TSTB    (R4)            ;DONE??
043366  100424                  BMI     9$              ;YES!!
043370  111401                  MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
043372  042701  177700          BIC     #^CIDXMSK,R1    ;CLEAR ANY SIGN EXTENSION
043376  060001                  ADD     R0,R1
043400  112403                  MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS
043402  042703  177700          BIC     #^CIDXMSK,R3    ;CLEAR ANY SIGN EXTENSION
043406  060203                  ADD     R2,R3
043410  011311                  MOV     (R3),(R1)       ;WRITE REGISTER
043412  000764                  BR      3$

043414  022626          5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
043416  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
043424  112776  000007  000016          MOVB    #7,@16(SP)      ;USER'S ERROR CALL
043432  162766  000002  000016  7$:     SUB     #2,16(SP)
```

```
10129
10130  043440                          9$:
10131  043440  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10132  043444  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10133  043450  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
10134  043452  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
10135  043454  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
10136  043456  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10137  043460  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10138  043462  000207                          RTS     PC              ;RETURN
10139
```

```
10140                                   .SBTTL  SIZE CLOCK SUBROUTINE
10141
10142  043464                           SIZCLK:
10143  043464  013746  000004                    MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10144  043470  013746  000006                    MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10145  043474  012737  043532  000004           MOV     #1$,ERRVEC      ;SET UP FOR BUS TIMEOUT
10146  043502  012737  000300  000006           MOV     #PR6,ERRVEC+2
10147  043510  012737  177546  001500           MOV     #177546,CLKADR  ;LOAD ADDRESSES FOR KW11-L
10148  043516  012737  000100  001502           MOV     #100,CLKVCT
10149  043524  005777  135750                   TST     @CLKADR         ;TEST FOR KW11-L PRESENT
10150  043530  000421                           BR      3$              ;YES - KW11-L IS PRESENT
10151  043532  022626                   1$:     CMP     (SP)+,(SP)+     ;RESTORE SP
10152  043534  012737  043564  000004           MOV     #2$,ERRVEC      ;SET UP FOR BUS TIMEOUT
10153  043542  012737  172540  001500           MOV     #172540,CLKADR  ;LOAD ADDRESSES FOR KW11-P CLOCK
10154  043550  012737  000104  001502           MOV     #104,CLKVCT
10155  043556  005777  135716                   TST     @CLKADR         ;TEST FOR KW11-P PRESENT
10156  043562  000404                           BR      3$              ;YES - KW11-P IS PRESENT
10157  043564  022626                   2$:     CMP     (SP)+,(SP)+     ;RESTORE SP
10158  043566  062766  000002  000004           ADD     #2,4(SP)        ;MOVE RETURN TO ERROR
10159  043574                           3$:
10160  043574  012637  000006                    MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10161  043600  012637  000004                    MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10162  043604  000207                           RTS     PC              ;RETURN TO USER
```

```
10163                                   .SBTTL  TIMEOUT SUBROUTINE
10164                                   ;
10165                                   ;THIS SUBROUTINE WAITS FOR GO TO RESET OR FOR A TIMEOUT GREATER THAN
10166                                   ;500 MS, WHICH EVER OCCURRS FIRST, AND THEN RETURNS.
10167                                   ;
10168                                   ;CALL:  JSR     PC,TIMOUT
10169                                   ;       ???                     RETURN HERE
10170
10171   043606                          TIMOUT:
10172   043606  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10173   043610  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10174   043612  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10175   043614  013746  000004                 MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10176   043620  013746  000006                 MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10177   043624  012737  043726  000004        MOV     #4$,ERRVEC      ;SETUP FOR BUS TIMEOUT - 04 TRAP
10178   043632  012737  000300  000006        MOV     #PR6,ERRVEC+2
10179   043640  013700  001276                 MOV     $BASE,R0        ;R0=RM03 ADDRESS
10180   043644  013701  001500                 MOV     CLKADR,R1       ;R1=CLOCK ADDRESS
10181   043650  012702  000037                 MOV     #31.,R2         ;R2=NUMBER OF CLOCK CYCLES
10182   043654  020127  172540          1$:    CMP     R1,#172540      ;KW11-P CLOCK??
10183   043660  001003                         BNE     2$              ;NO!!
10184   043662  012761  000001  000002        MOV     #1,2(R1)        ;SET COUNTER
10185   043670  012711  000005          2$:    MOV     #BIT2!BIT0,(R1) ;START COUNTER
10186
10187   043674  016046  000000          3$:    MOV     RMCS1(R0),-(SP) ;GET STATUS
10188   043700  042716  177576                 BIC     #^C<RDY!GO>,(SP)
10189   043704  022726  000200                 CMP     #RDY,(SP)+      ;RDY=1,GO=0??
10190   043710  001420                         BEQ     5$              ;YES!!
10191   043712  032711  000200                 BIT     #BIT7,(R1)      ;TIMER DONE??
10192   043716  001766                         BEQ     3$              ;NO!!
10193   043720  005302                         DEC     R2              ;DEC NUMBER OF CYCLES
10194   043722  001354                         BNE     1$              ;CONTINUE IF NOT DONE
10195   043724  000412                         BR      5$
10196   043726  022626          4$:    CMP     (SP)+,(SP)+     ;ADJUST STACK
10197   043730  062766  000004  000012        ADD     #4,12(SP)       ;MOVE SP TO USER'S CALL
10198   043736  112776  000007  000012        MOVB    #7,@12(SP)      ;WRITE ERROR NUMBER
10199   043744  162766  000002  000012        SUB     #2,12(SP)
10200
10201   043752                          5$:
10202   043752  012637  000006                 MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10203   043756  012637  000004                 MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10204   043762  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
10205   043764  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
10206   043766  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
10207   043770  000207                         RTS     PC              ;RETURN TO USER
10208
```

```
10209                                   .SBTTL  PRIMARY ERROR CHECK SUBROUTINE
10210                                   ;
10211                                   ;THE PURPOSE OF THIS SUBROUTINE IS TO VERIFY THAT STAUTS IS VALID AND
10212                                   ;THAT FURTHER ERROR AND STATUS CHECKING SHOULD BE PERFORMED.  THE
10213                                   ;FOLLOWING CHECKS ARE MADE:
10214                                   ;
10215                                   ;       .CURRECT UNIT IS SELECTED, I.E., THE UNIT SELECT BITS OF RMCS2
10216                                   ;(BITS 0-2) EQUAL THE UNIT BEING TESTED;
10217                                   ;
10218                                   ;       .SELECTED UNIT IS AVAILABLE, I.E., DVA (BIT 11 OF RMCS1) IS SET
10219                                   ;AND NED (BIT 12 OF RMCS2) IS RESET;
10220                                   ;
10221                                   ;       .LAST COMMAND WAS COMPLETED, I.E., THE MASSBUS CONTROLLER IS
10222                                   ;READY (BIT 7 OF RMCS1) AND THE GO BIT IS RESET (BIT 0 OF RMCS1) OR THE
10223                                   ;DRIVE READY BIT (BIT 7 OF RMDS) IS SET.
10224                                   ;       .NO PARITY ERROR OCCURRED WHEN READING REMOTE REGISTERS,
10225                                   ;I.E., MCPE = 0.
10226                                   ;       .NO PARITY ERROR OCCURRED WHEN WRITING REMOTE REGISTERS,
10227                                   ;I.E., PAR = 0, OR, PAR = DPE = 1
10228                                   ;
10229                                   ;
10230                                   ;THE SUBROUTINE ASSUMES THAT:
10231                                   ;
10232                                   ;       .STATUS HAS BEEN STORED IN THE REGISTER INPUT BUFFER,
10233                                   ;IN PARTICULAR, RMCS1, RMCS2 AND RMDS HAVE BEEN STORED IN THEIR
10234                                   ;CORRESPONDING LOCATIONS OF THE "GET" BUFFER.
10235                                   ;
10236                                   ;       .($UNIT) CONTAINS THE DRIVE NUMBER
10237                                   ;
10238                                   ;THE SUBROUTINE IS CALLED AS FOLLOWS:
10239                                   ;
10240                                   ;(1)    JSR     PC,PRIERR
10241                                   ;       BR      ???             RETURN HERE IF NO ERROR
10242                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10243                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10244                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10245                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10246
10247   043772                          PRIERR:
10248
10249                                   ;CLEAR USER'S ERROR CALL
10250   043772  062716  000004                 ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
10251   043776  105076  000000                 CLRB    @(SP)           ;CLEAR ERROR NUMBER
10252   044002  162716  000004                 SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10253
10254
10255                                   ;REPORT AN ERROR IF THE WRONG UNIT IS SELECTED
10256   044006  013737  001336  001142         MOV     RMCS2I,$BDDAT   ;CORRECT UNIT SELECTED??
10257   044014  042737  177770  001142         BIC     #^CUNTMSK,$BDDAT
10258   044022  013737  001234  001140         MOV     $UNIT,$GDDAT    ;GOOD DATA FOR TYPEOUT
10259   044030  042737  177770  001140         BIC     #^CUNTMSK,$GDDAT
10260   044036  123737  001140  001142         CMPB    $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
10261                                                                   ;DRIVE NUMBERS
10262   044044  001415                         BEQ     1$              ;YES!!
10263   044046  062716  000004                 ADD     #4,(SP)
10264   044052  112776  000001  000000         MOVB    #1,@(SP)        ;ERROR 1
```

```
044060  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
044064  004736                          JSR     PC,@(SP)+        ;REPORT WRONG UNIT SELECTED
044066  162716  000010                  SUB     #10,(SP)         ;RESTORE (SP)
044072  000240                          NOP
044074  000137  044614                  JMP     10$              ;SKIP OTHER CHECKS
044100                          1$:

                                ;REPORT AN ERROR IF THE DEVICE IS NOT AVAILABLE OR IF
                                ;THE DEVICE IS NONEXISTANT
044100  032737  004000  001326          BIT     #DVA,RMCS1I      ;DEVICE AVAILABLE??
044106  001045                          BNE     5$               ;YES!!
044110  013737  001326  001140          MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
044116  052737  004000  001140          BIS     #DVA,$GDDAT
044124  013737  001326  001142          MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
044132  062716  000004                  ADD     #4,(SP)
044136  112776  000002  000000          MOVB    #2,@(SP)         ;ERROR #2
044144  032737  010000  001336          BIT     #NED,RMCS2I      ;WAS NED SET??
044152  001414                          BEQ     2$               ;NO!!
044154  013737  001336  001140          MOV     RMCS2I,$GDDAT    ;EXPECTED STATUS
044162  013737  001336  001142          MOV     RMCS2I,$BDDAT    ;RECEIVED STATUS
044170  042737  010000  001140          BIC     #NED,$GDDAT
044176  112776  000003  000000          MOVB    #3,@(SP)         ;YES - CHANGE ERROR NUMBER
044204  162716  000002          2$:     SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
044210  004736                          JSR     PC,@(SP)+        ;REPORT DEVICE NOT AVAILABLE
044212  162716  000010                  SUB     #10,(SP)         ;RESTORE (SP)
044216  000240                          NOP
044220  000575                          BR      10$              ;SKIP OTHER CHECKS
044222                          5$:

                                ;REPORT AN ERROR IF MASSBUS CONTROLLER IS NOT READY
044222  032737  000200  001326          BIT     #RDY,RMCS1I      ;CONTROLLER READY??
044230  001030                          BNE     7$               ;YES!!
044232  013737  001326  001140          MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
044240  052737  000200  001140          BIS     #RDY,$GDDAT
044246  042737  160001  001140          BIC     #SC!TRE!MCPE!GO,$GDDAT
044254  013737  001326  001142          MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
044262  062716  000004                  ADD     #4,(SP)
044266  112776  000004  000000          MOVB    #4,@(SP)         ;ERROR #4
044274  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
044300  004736                          JSR     PC,@(SP)+        ;REPORT CONTROLLER NOT READY
044302  162716  000010                  SUB     #10,(SP)         ;RESTORE (SP)
044306  000240                          NOP
044310  000541                          BR      10$              ;SKIP OTHER CHECKS
044312                          7$:

                                ;REPORT AN ERROR IF GO IS NOT ZERO AND DRY IS NOT ONE
044312  032737  000001  001326          BIT     #GO,RMCS1I       ;GO RESET??
044320  001431                          BEQ     8$               ;YES!!
044322  032737  000200  001340          BIT     #DRY,RMDSI       ;DRIVE READY??
044330  001025                          BNE     8$               ;YES!!
044332  013737  001326  001140          MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
044340  042737  160001  001140          BIC     #SC!TRE!MCPE!GO,$GDDAT
044346  013737  001326  001142          MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
044354  062716  000004                  ADD     #4,(SP)
044360  112776  000005  000000          MOVB    #5,@(SP)         ;ERROR #5
044366  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
```

```
10321  044372  004736                          JSR     PC,@(SP)+       ;REPORT DRIVE NOT READY
10322  044374  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
10323  044400  000240                          NOP
10324  044402  000504                          BR      10$
10325  044404                          8$:
10326
10327                                  ;REPORT AN ERROR IF THE RH CONTROLLER DETECTED BAD
10328                                  ;PARITY ON THE MASSBUS CONTROL BUS
10329  044404  032737  020000  001326            BIT     #MCPE,RMCS1I    ;PARITY ERROR ??
10330  044412  001425                          BEQ     9$              ;NO!!
10331  044414  013737  001326  001140          MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
10332  044422  042737  160001  001140          BIC     #SC!TRE!MCPE!GO,$GDDAT
10333  044430  013737  001326  001142          MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
10334  044436  062716  000004                  ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
10335  044442  112776  000013  000000          MOVB    #13,@(SP)       ;ERROR #47
10336  044450  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10337  044454  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10338  044456  162716  000010                  SUB     #10,(SP)        ;RESTORE STACK
10339  044462  000240                          NOP
10340  044464  000453                          BR      10$
10341  044466                          9$:
10342
10343                                  ;REPORT AN ERROR IF THE RM03 DETECTED A CONTROL BUS PARITY ERROR
10344  044466  032737  000010  001342            BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
10345  044474  001451                          BEQ     11$             ;NO!!
10346  044476  032737  000010  001370          BIT     #DPE,RMER2I     ;WAS IT THE CONTROL BUS??
10347  044504  001045                          BNE     11$             ;NOT SURE!!
10348  044506  032737  000010  001412          BIT     #PAR,RMER10     ;DID TEST SET PAR ??
10349  044514  001413                          BEQ     93$             ;NO!!
10350  044516  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10351  044520  012700  001533                  MOV     #PUTINX,R0      ;R0 POINTS TO INDEX TABLE
10352  044524  122710  000014          91$:    CMPB    #RMER1,(R0)     ;SEARCH TABLE FOR RMER1
10353  044530  001002                          BNE     92$
10354  044532  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10355  044534  000431                          BR      11$             ;PAR WAS SET BY TEST
10356  044536  105720                  92$:    TSTB    (R0)+           ;END OF TABLE??
10357  044540  100371                          BPL     91$             ;NO!!
10358  044542  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10359  044544  013737  001342  001140  93$:    MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10360  044552  042737  000010  001140          BIC     #PAR,$GDDAT
10361  044560  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10362  044566  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10363  044572  112776  000050  000000          MOVB    #50,@(SP)       ;WRITE THE ERROR NUMBER
10364  044600  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10365  044604  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR
10366  044606  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10367  044612  000240                          NOP
10368  044614  062716  000010          10$:    ADD     #10,(SP)                ;RETURN TO ERROR
10369  044620  000240                  11$:    NOP                     ;RETURN TO NO ERROR
10370  044622  000207                          RTS     PC
10371
```

```
                                .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.
                                ;
044624                          SECERR:
                                ;
                                ;;*****************************************************************
                                ;STORE FUNCTION CODE AND CLEAR USER'S ERROR NUMBER
044624  013737  001376  050464          MOV     RMCS10,5155$    ;STORE FUNCTION CODE
044632  042737  177701  050464          BIC     #^C<F0!F1!F2!F3!F4>,5155$
044640  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
044644  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
044650  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
044654  032737  000200  001340          BIT     #DRY,RMDSI      ;DRIVE READY??
044662  001024                          BNE     5$              ;YES!!
044664  013737  001340  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
044672  042737  177577  001142          BIC     #^CDRY,$BDDAT
044700  012737  000200  001140          MOV     #DRY,$GDDAT     ;GOOD DATA FOR TYPEOUT
044706  062716  000004                  ADD     #4,(SP)
044712  112776  000010  000000          MOVB    #10,@(SP)       ;ERROR NUMBER
044720  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
044724  004736                          JSR     PC,@(SP)+       ;REPORT NOT READY
044726  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP) TO ERROR N
044732  000240                          NOP
                                ;
                                ;REPORT ERROR IF GO BIT IS NOT RESET
044734  032737  000001  001326  5$:     BIT     #GO,RMCS1I      ;GO BIT RESET??
044742  001423                          BEQ     10$             ;YES!!
044744  013737  001326  001142          MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
044752  042737  177776  001142          BIC     #^CGO,$BDDAT
044760  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
044764  062716  000004                  ADD     #4,(SP)
```

```
10426  045770  112776  000011  000000          MOVB    #11,@(SP)       ;ERROR NUMBER
10427  045776  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10430  046002  004736                          JSR     PC,@(SP)+       ;REPORT DEVICE NOT AVAILABLE
10431  046004  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10432  046010  000240                          NOP
10433
10434                                  ;REPORT ERROR IF FUNCTION CODE READ FROM DEVICE IS NOT CORRECT
10435  046012  013737  001326  001142  10$:     MOV     RMCS1I,$BDDAT   ;IS FUNCTION CODE CORRECT??
10436  046020  042737  177701  001142          BIC     #^C76,$BDDAT
10437  046026  013737  050464  001140          MOV     515$,$GDDAT     ;EXPECTED FUNCTION CODE
10438  046034  023737  001142  001140          CMP     $BDDAT,$GDDAT
10439  046042  001413                          BEQ     15$             ;YES!!
10440  046044  062716  000004                  ADD     #4,(SP)
10441  046050  112776  000012  000000          MOVB    #12,@(SP)       ;ERROR NUMBER
10442  046056  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10443  046062  004736                          JSR     PC,@(SP)+       ;REPORT WRONG FUNCTION CODE
10444  046064  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10445  046070  000240                          NOP
10446  046072                          15$:
10447                                  ;REPORT AN ERROR IF COMPOSITE ERROR IS SET AND NO OTHER
10448                                  ;ERRORS ARE SET, OR IF COMPOSITE ERROR IS NOT SET AND
10449                                  ;OTHER ERRORS ARE SET
10450  046072  005037  001140                  CLR     $GDDAT          ;EXPECT "ERR"=0
10451  046076  005737  001342                  TST     RMER1I          ;IS RMER1 =0??
10452  046102  001003                          BNE     20$             ;NO!!
10453  046104  005737  001370                  TST     RMER2I          ;IS RMERZ=0??
10454  046110  001403                          BEQ     25$             ;YES!!
10455  046112  052737  040000  001140  20$:     BIS     #ERR,$GDDAT     ;"ERR" SOULD BE SET
10456  046120  013737  001340  001142  25$:     MOV     RMDSI,$BDDAT
10457  046126  042737  137777  001142          BIC     #^CERR,$BDDAT
10458  046134  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "ERR" OK??
10459  046142  001412                          BEQ     30$             ;YES!!
10460  046144  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
10461  046150  112776  000047  000000          MOVB    #47,@(SP)       ;WRITE ERROR NUMBER
10462  046156  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10463  046162  004736                          JSR     PC,@(SP)+       ;REPORT INVALID COMP ERROR
10464  046164  162716  000010                  SUB     #10,(SP)
10465
10466                                  ;REPORT AN ERROR IF "TRE" IS SET AND NONE OF THE BITS WHICH SET
10467                                  ;TRE IS SET, OR IF TRE IS NOT SET AND ONE OR MORE BITS WHICH
10468                                  ;SET TRE IS SET
10469  046170  005037  001140          30$:    CLR     $GDDAT          ;EXPECT "TRE" =0
10470  046174  013746  001336                  MOV     RMCS2I,-(SP)    ;WAS DLT, WCE, UPE, NED, NEM
10471  046200  042726  000377                  BIC     #377,(SP)+      ;PGE, MXF OR MDPE SET
10472  046204  001010                          BNE     35$             ;YES!!
10473  046206  032737  040000  001340          BIT     #ERR,RMDSI      ;WAS EXCEPTION RECEIVED??
10474  046214  001407                          BEQ     40$             ;NO!!
10475  046216  022737  000030  050464          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
10476  046224  103003                          BHIS    40$             ;NO!!
10477  046226  052737  040000  001140  35$:     BIS     #TRE,$GDDAT     ;"TRE" SHOULD BE SET
10478  046234  013737  001326  001142  40$:     MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
10479  046242  042737  137777  001142          BIC     #^CTRE,$BDDAT
10480  046250  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "TRE" OK??
10481  046256  001413                          BEQ     45$             ;YES!!
10482  046260  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10483  046264  112776  000014  000000          MOVB    #14,@(SP)       ;WRITE ERROR NUMBER
```

```
10484  045272  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10485  045276  004736                          JSR     PC,@(SP)+        ;REPORT TRE ERROR
10486  045300  162716  000010                  SUB     #10,(SP)         ;RESTORE (SP)
10487  045304  000240                          NOP
10488  045306                          45$:
10489
10490                                  ;;*****************************************************************
10491                                  ;USING THE FUNCTION CODE TABLE, CHECK FOR THE FOLLOWING ERRORS:
10492                                  ;       .STATUS BITS NOT SET THAT SHOULD BE SET, E.G., ATA AND ILF
10493                                  ;       .STATUS BITS SET THAT SHOULD NOT BE SET, E.G., WCE AND ECH
10494                                  ;NOTE THAT SOME ERROR BITS ARE CONDITIONAL ON THE COMMAND AND OTHER
10495                                  ;STATUS CONDITIONS, E.G., WRITE LOCK ERROR SHOULD ONLY BE SET IF
10496                                  ;WRITE LOCK IS ON AND THE COMMAND IS A WRITE.
10497
10498                                  ;GET AND STORE THE ENTRY FROM THE FUNCTION CODE TABLE
10499  045306  010046                          MOV     R0,-(SP)         ;;PUSH R0 ON STACK
10500  045310  013700  050464                  MOV     515$,R0          ;R0 = FUNCTION CODE
10501  045314  016037  067646  050456          MOV     FNCDTB(R0),500$  ;STORE ENTRY
10502  045322  012600                          MOV     (SP)+,R0         ;;POP STACK INTO R0
10503
10504                                  ;REPORT AN ERROR IF AN UNEXPECTED ATTENTION OCCURRED OR IF
10505                                  ;ATA IS NOT SET AND SHOULD BE SET.
10506  045324  013737  050456  001140          MOV     500$,$GDDAT      ;GET EXPECTED ATA STATUS
10507  045332  032737  040000  001340          BIT     #ERR,RMDSI       ;IS COMPOSITE ERROR SET ??
10508  045340  001403                          BEQ     50$              ;NO !!
10509  045342  052737  100000  001140          BIS     #ATA,$GDDAT      ;EXPECT AN ATTENTION
10510  045350  042737  077777  001140  50$:    BIC     #^CATA,$GDDAT    ;STRIP DONT CARES
10511  045356  013737  001340  001142          MOV     RMDSI,$BDDAT     ;GET RECEIVED ATA
10512  045364  042737  077777  001142          BIC     #^CATA,$BDDAT    ;STRIP DONT CARES
10513  045372  023737  001140  001142          CMP     $GDDAT,$BDDAT    ;IS ATA OK ??
10514  045400  001413                          BEQ     55$              ;YES !!
10515  045402  062716  000004                  ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
10516  045406  112776  000006  000000          MOVB    #6,@(SP)         ;LOAD ERROR # IN CALL
10517  045414  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
10518  045420  004736                          JSR     PC,@(SP)+        ;REPORT ERROR
10519  045422  162716  000010                  SUB     #10,(SP)         ;RESTORE SP
10520  045426  000240                          NOP
10521  045430                          55$:
10522
10523                                  ;REPORT ERROR IF ILF IS INCORRECT, I.E., IF ILF DOES NOT COMPARE
10524                                  ;WITH FUNCTION CODE TABLE
10525  045430  013737  050456  001140          MOV     500$,$GDDAT      ;GET EXPECTED ILF
10526  045436  042737  177776  001140          BIC     #^CILF,$GDDAT    ;CLEAR ALL OTHER BITS
10527  045444  013737  001342  001142          MOV     RMER1I,$BDDAT    ;GET RECEIVED ILF
10528  045452  042737  177776  001142          BIC     #^CILF,$BDDAT    ;CLEAR ALL OTHER BITS
10529  045460  023737  001140  001142          CMP     $GDDAT,$BDDAT    ;IS ILF OK ??
10530  045466  001412                          BEQ     60$              ;YES !!
10531  045470  062716  000004                  ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
10532  045474  112776  000254  000000          MOVB    #254,@(SP)       ;WRITE ERROR NUMBER IN CALL
10533  045502  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
10534  045506  004736                          JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
10535  045510  162716  000010                  SUB     #10,(SP)         ;MOVE SP TO NO ERROR
10536  045514  005037  001140          60$:    CLR     $GDDAT           ;CLEAR EXPECTED STATUS
10537
10538                                  ;REPORT AN ERROR IF WCE IS SET AND SHOULD NOT BE SET
10539  045520  013746  050456                  MOV     500$,-(SP)       ;GET WCE STATUS ENABLE
```

```
10540  045524  052716  137777            BIS     #↑CWCE,(SP)     ;SET ALL OTHER BITS
10541  045530  013737  001336  001142    MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
10542  045536  042637  001142            BIC     (SP)+,$BDDAT    ;CLEAR WCE IF ENABLED
10543  045542  001412                    BEQ     90$             ;BRANCH IF WCE OK
10544  045544  062716  000004            ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10545  045550  112776  000026  000000    MOVB    #26,@(SP)       ;WRITE ERROR NUMBER
10546  045556  162716  000002            SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10547  045562  004736                    JSR     PC,@(SP)+       ;REPORT ERROR
10548  045564  162716  000010            SUB     #10,(SP)        ;RESTORE ERROR
10549  045570                    90$:
10550
10551                            ;REPORT ERROR IF OPI STATUS IS SET AND SHOULD NOT BE SET
10552  045570  013746  050456            MOV     500$,-(SP)      ;GET OPI STATUS ENABLE
10553  045574  052716  157777            BIS     #↑COPI,(SP)     ;SET ALL OTHER BITS
10554  045600  013737  001342  001142    MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10555  045606  042637  001142            BIC     (SP)+,$BDDAT    ;CLEAR OPI IF ENABLED
10556  045612  001412                    BEQ     100$            ;BRANCH IF OPI OK
10557  045614  062716  000004            ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10558  045620  112776  000164  000000    MOVB    #164,@(SP)      ;WRITE ERROR NUMBER IN CALL
10559  045626  162716  000002            SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10560  045632  004736                    JSR     PC,@(SP)+       ;REPORT ERROR
10561  045634  162716  000010            SUB     #10,(SP)        ;RESTORE SP
10562  045640                    100$:
10563
10564                            ;REPORT ERROR IF IVC IS SET AND IS NOT ENABLED OR IF IVC IS
10565                            ;SET AND VV IS NOT RESET
10566  045640  013746  050456            MOV     500$,-(SP)      ;GET IVC STATUS ENABLE
10567  045644  032737  000100  001340    BIT     #VV,RMDSI       ;IS VV SET
10568  045652  001402                    BEQ     105$            ;NO !!
10569  045654  042716  010000            BIC     #IVC,(SP)       ;YES - IVC SHOULD BE 0
10570  045660  052716  167777    105$:   BIS     #↑CIVC,(SP)     ;SET ALL OTHER BITS
10571  045664  013737  001370  001142    MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
10572  045672  042637  001142            BIC     (SP)+,$BDDAT    ;CLEAR IVC IF ENABLED
10573  045676  001412                    BEQ     110$            ;BRANCH IF IVC OK
10574  045700  062716  000004            ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10575  045704  112776  000165  000000    MOVB    #165,@(SP)      ;WRITE ERROR NUMBER IN CALL
10576  045712  162716  000002            SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10577  045716  004736                    JSR     PC,@(SP)+       ;REPORT ERROR
10578  045720  162716  000010            SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10579  045724                    110$:
10580
10581                            ;BIT 11 (WLE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10582                            ; ALL WRITE ERRORS, I.E.,
10583                            ;       RMER1 - WLE, WCF
10584                            ;       RMER2 - DPE
10585                            ;       RMCS2 - UPE.
10586                            ;EACH OF THESE ERRORS IS CHECKED TO SEE IF AN ERROR IS SET WHEN THE
10587                            ;WRITE ERROR ENABLE BIT IS RESET.
10588
10589                            ;REPORT AN ERROR IF WLE IS SET AND WRITE ERRORS ARE NOT ENABLED, OR IF
10590                            ;THE DRIVE IS NOT WRITE PROTECTED
10591  045724  012746  177777            MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ENABLED
10592  045730  032737  004000  050456    BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
10593  045736  001404                    BEQ     115$            ;NO !!
10594  045740  032737  004000  001340    BIT     #WRL,RMDSI      ;IS THE DRIVE WRITE PROTECTED ??
10595  045746  001002                    BNE     120$            ;YES !!
```

```
045750  042716  004000          115$:   BIC     #WLE,(SP)           ;RESET WLE ENABLE
045754  013737  001342  001142  120$:   MOV     RMER1I,$BDDAT       ;GET RECEIVED STATUS
045762  042637  001142                  BIC     (SP)+,$BDDAT        ;CLEAR WLE IF ENABLED
045766  001412                          BEQ     125$                ;BRANCH IF WLE OK
045770  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USERS ERROR CALL
045774  112776  000023  000000          MOVB    #23,@(SP)           ;WRITE ERROR NUMBER IN CALL
046002  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO ERROR RETURN
046006  004736                          JSR     PC,@(SP)+           ;REPORT ERROR AND RETURN
046010  162716  000010                  SUB     #10,(SP)            ;RESTORE SP TO NO ERROR
046014                          125$:

                                ;REPORT ERROR IF WCF IS SET AND WRITE ERRORS ARE NOT ENABLED
046014  012746  177777                  MOV     #-1,-(SP)           ;ASSUME WRITE ERRORS ENABLED
046020  032737  004000  050456          BIT     #WLE,500$           ;ARE WRITE ERRORS ENABLED ??
046026  001002                          BNE     130$                ;YES !!
046030  042716  000040                  BIC     #WCF,(SP)           ;DISABLE WCF ERROR
046034  013737  001342  001142  130$:   MOV     RMER1I,$BDDAT       ;GET RECEIVED STATUS
046042  042637  001142                  BIC     (SP)+,$BDDAT        ;RESET WCF IF ENABLED
046046  001412                          BEQ     135$                ;BRANCH IF WCF OK
046050  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USERS ERROR CALL
046054  112776  000025  000000          MOVB    #25,@(SP)           ;WRITE ERROR NUMBER IN CALL
046062  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO ERROR RETURN
046066  004736                          JSR     PC,@(SP)+           ;REPORT ERROR
046070  162716  000010                  SUB     #10,(SP)            ;RESTORE SP TO NO ERROR
046074                          135$:

                                ;REPORT ERROR IF DPE IS SET AND WRITE ERRORS ARE NOT ENABLED
046074  012746  177777                  MOV     #-1,-(SP)           ;ASSUME WRITE ERRORS ARE ENABLED
046100  032737  004000  050456          BIT     #WLE,500$           ;ARE WRITE ERRORS ENABLED ??
046106  001002                          BNE     140$                ;YES !!
046110  042716  000010                  BIC     #DPE,(SP)           ;RESET DPE ENABLE
046114  013737  001370  001142  140$:   MOV     RMER2I,$BDDAT       ;GET RECEIVED STATUS
046122  042637  001142                  BIC     (SP)+,$BDDAT        ;RESET DPE IF ENABLED
046126  001412                          BEQ     145$                ;BRANCH IF DPE OK
046130  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USERS ERROR CALL
046134  112776  000040  000000          MOVB    #40,@(SP)           ;WRITE ERROR NUMBER IN CALL
046142  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO ERROR RETURN
046146  004736                          JSR     PC,@(SP)+           ;REPORT ERROR
046150  162716  000010                  SUB     #10,(SP)            ;RESTORE SP TO NO ERROR
046154                          145$:

                                ;REPORT AN ERROR IF UPE IS SET AND WRITE ERRORS ARE NOT ENABLED
046154  012746  177777                  MOV     #-1,-(SP)           ;ASSUME WRITE ERRORS ARE ENABLED
046160  032737  004000  050456          BIT     #WLE,500$           ;ARE WRITE ERRORS ENABLED ??
046166  001002                          BNE     150$                ;YES !!
046170  042716  020000                  BIC     #UPE,(SP)           ;DISABLE UPE ERROR
046174  013737  001336  001142  150$:   MOV     RMCS2I,$BDDAT       ;GET RECEIVED STATUS
046202  042637  001142                  BIC     (SP)+,$BDDAT        ;RESET UPE IF ENABLED
046206  001412                          BEQ     155$                ;BRANCH IF UPE OK
046210  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USERS ERROR CALL
046214  112776  000024  000000          MOVB    #24,@(SP)           ;WRITE ERROR NUMBER IN CALL
046222  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO ERROR RETURN
046226  004736                          JSR     PC,@(SP)+           ;REPORT ERROR AND RETURN
046230  162716  000010                  SUB     #10,(SP)            ;MOVE SP TO NO ERROR
046234                          155$:
```

```
10652
10653  046234  013746  050456                  ;REPORT AN ERROR IF IAE IS SET AND IS NOT ENABLED
10654  046240  052716  175777                          MOV     500$,-(SP)      ;GET IAE ENABLE
10655  046244  013737  001342  001142                  BIS     #^CIAE,(SP)     ;SET ALL OTHER BITS
10656  046252  042637  001142                          MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10657  046256  001412                                  BIC     (SP)+,$BDDAT    ;CLEAR IAE IF ENABLED
10658  046260  062716  000004                          BEQ     160$            ;BRANCH IF IAE IS OK
10659  046264  112776  000166  000000                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10660  046272  162716  000002                          MOVB    #166,@(SP)      ;WRITE ERROR NUMBER
10661  046276  004736                                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10662  046300  162716  000010                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10663  046304                                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
                                               160$:
10664
10665                                          ;BIT 09 (AOE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10666                                          ; ALL READ/WRITE ERRORS, I.E.,
10667                                          ;       RMCS1 - TRE
10668                                          ;       RMCS2 - DLT,NEM,MXF
10669                                          ;       RMDS  - LBT
10670                                          ;       RMER1 - AOE
10671                                          ;NOTE:
10672                                          ;       LBT IS NOT CHECKED BECAUSE IT ONLY RESETS WHEN THE DESIRED
10673                                          ;CYLINDER REGISTER IS WRITTEN
10674                                          ;NOTE:
10675                                          ;       AOE CANNOT BE SET IF LBT IS NOT ALSO SET
10676                                          ;NOTE:
10677                                          ;       TRE IS CHECKED AS A FUNCTION OF OTHER ERROR CONDITONS ABOVE
10678
10679
10680                                          ;REPORT AN ERROR IF DLT IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10681  046304  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10682  046310  032737  001000  050456                  BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
10683  046316  001002                                  BNE     165$            ;YES !!
10684  046320  042716  100000                          BIC     #DLT,(SP)       ;RESET DLT ENABLE
10685  046324  013737  001336  001142          165$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10686  046332  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR DLT IF ENABLED
10687  046336  001412                                  BEQ     170$            ;BRANCH IF DLT IS OK
10688  046340  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10689  046344  112776  000032  000000                  MOVB    #32,@(SP)       ;WRITE ERROR NUMBER IN CALL
10690  046352  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10691  046356  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10692  046360  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10693  046364                                 170$:
10694
10695                                          ;REPORT ERROR IF NEM IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10696  046364  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10697  046370  032737  001000  050456                  BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
10698  046376  001002                                  BNE     175$            ;YES !!
10699  046400  042716  004000                          BIC     #NEM,(SP)       ;DISABLE NEM
10700  046404  013737  001336  001142          175$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10701  046412  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR NEM IF ENABLED
10702  046416  001412                                  BEQ     180$            ;BRANCH IF NEM IS OK
10703  046420  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10704  046424  112776  000167  000000                  MOVB    #167,@(SP)      ;WRITE ERROR NUMBER IN CALL
10705  046432  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10706  046436  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10707  046440  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
```

```
10708  046444                                  180$:
10709
10710                                          ;REPORT ERROR IF MXF IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10711  046444  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10712  046450  032737  001000  050456                  BIT     #AOE,500$       ;ARE DATA ERRORS ENABLED ??
10713  046456  001002                                  BNE     185$            ;YES !!
10714  046460  042716  001000                          BIC     #MXF,(SP)       ;DISABLE MXF ERROR
10715  046464  013737  001336  001142  185$:          MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10716  046472  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR MXF IF ENABLED
10717  046476  001412                                  BEQ     190$            ;BRANCH IF MXF IS OK
10718  046500  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10719  046504  112776  000033  000000                  MOVB    #33,@(SP)       ;WRITE ERROR NUMBER IN CALL
10720  046512  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10721  046516  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10722  046520  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10723  046524                                  190$:
10724
10725                                          ;REPORT ERROR IF AOE IS SET AND DATA ERRORS ARE NOT ENABLED
10726  046524  012746  177777                          MOV     #-1,-(SP)       ;ASSUME DATA ERRORS ARE ENABLED
10727  046530  032737  001000  050456                  BIT     #AOE,500$       ;ARE DATA ERRORS EANBLED ??
10728  046536  001404                                  BEQ     191$            ;NO !!
10729  046540  032737  002000  001340                  BIT     #LBT,RMDSI      ;IS LBT ALSO SET ??
10730  046546  001002                                  BNE     195$            ;YES !!
10731  046550  042716  001000                  191$:   BIC     #AOE,(SP)       ;DISABLE AOE
10732  046554  013737  001342  001142  195$:          MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10733  046562  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR AOE IF ENABLED
10734  046566  001412                                  BEQ     200$            ;BRANCH IF AOE IS OK
10735  046570  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10736  046574  112776  000020  000000                  MOVB    #20,@(SP)       ;WRITE ERROR NUMBER
10737  046602  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10738  046606  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10739  046610  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10740  046614                                  200$:
10741
10742                                          ;BIT 07 (HCE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10743                                          ;HEADER ERRORS, I.E.,
10744                                          ;       RMER1 - HCRC,HCE,FER
10745                                          ;       RMER2 - BSE
10746
10747                                          ;RESET THE ENABLING BIT (HCE) IF HEADER COMPARE INHIBIT IS SET
10748  046614  032737  002000  001360                  BIT     #HCI,RMOFI      ;IS HCI SET ??
10749  046622  001403                                  BEQ     201$            ;NO !!
10750  046624  042737  000200  050456                  BIC     #HCE,500$       ;YES - DISABLE ALL HEADER ERRORS
10751  046632                                  201$:
10752
10753                                          ;REPORT AN ERROR IF HCRC IS SET AND HEADER ERRORS ARE NOT ENABLED
10754  046632  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10755  046636  032737  000200  050456                  BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
10756  046644  001002                                  BNE     205$            ;YES !!
10757  046646  042716  000400                          BIC     #HCRC,(SP)      ;DISABLE HCRC
10758  046652  013737  001342  001142  205$:          MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10759  046660  042637  001142                          BIC     (SP)+,$BDDAT    ;RESET HCRC IF ENABLED
10760  046664  001412                                  BEQ     210$            ;BRANCH IF HCRC IS OK
10761  046666  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10762  046672  112776  000035  000000                  MOVB    #35,@(SP)       ;WRITE ERROR NUMBER IN CALL
10763  046700  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
```

```
10764  046704  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10765  046706  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10766  046712                        210$:
10767
10768                                ;REPORT ERROR IF HCE IS SET AND HEADER ERRORS ARE NOT ENABLED
10769  046712  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10770  046716  032737  000200  050456        BIT     #HCE,500$       ;ARE ERRORS ENABLED ??
10771  046724  001002                        BNE     215$            ;YES !!
10772  046726  042716  000200                BIC     #HCE,(SP)       ;DISABLE HCE
10773  046732  013737  001342  001142 215$:  MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10774  046740  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR HCE IF ENABLED
10775  046744  001412                        BEQ     220$            ;BRANCH IF HCE IS OK
10776  046746  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10777  046752  112776  000036  000000        MOVB    #36,@(SP)       ;WRITE ERROR NUMBER IN CALL
10778  046760  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10779  046764  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10780  046766  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10781  046772                        220$:
10782
10783                                ;REPORT ERROR IF FER IS SET AND HEADER ERRORS ARE NOT ENABLED
10784  046772  012746  177777                MOV     #-1,-(SP)       ;ASSUME FER IS ENABLED
10785  046776  032737  000200  050456        BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
10786  047004  001002                        BNE     225$            ;YES !!
10787  047006  042716  000020                BIC     #FER,(SP)       ;DISABLE FER
10788  047012  013737  001342  001142 225$:  MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10789  047020  042637  001142                BIC     (SP)+,$BDDAT    ;RESET FER IF ENABLED
10790  047024  001412                        BEQ     230$            ;BRANCH IF FER OK
10791  047026  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10792  047032  112776  000037  000000        MOVB    #37,@(SP)       ;WRITE ERROR NUMBER IN CALL
10793  047040  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10794  047044  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10795  047046  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10796  047052                        230$:
10797
10798                                ;REPORT ERROR IF BSE IS SET AND HEADER ERRORS ARE NOT ENABLED
10799  047052  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10800  047056  032737  000200  050456        BIT     #HCE,500$       ;ARE THEY ENABLED ??
10801  047064  001002                        BNE     235$            ;YES !!
10802  047066  042716  100000                BIC     #BSE,(SP)       ;DISABLE BSE
10803  047072  013737  001370  001142 235$:  MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
10804  047100  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR BSE IF ENABLED
10805  047104  001412                        BEQ     240$            ;BRANCH IF BSE OK
10806  047106  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10807  047112  112776  000354  000000        MOVB    #354,@(SP)      ;WRITE ERROR NUMBER
10808  047120  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10809  047124  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10810  047126  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10811  047132                        240$:
10812
10813                                ;BIT 06 OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR DATA
10814                                ;FIELD ERRORS, I.E.,
10815                                ;       RMCS2 - MDPE
10816                                ;       RMER1 - DCK,ECH
10817                                ;NOTE:
10818                                ;       ECH CANNOT SET UNLESS IT IS ENABLED AND ECI IS RESET AND
10819                                ;DCK IS SET.
```

```
10820
10821                                   ;REPORT ERROR IF MDPE IS SET AND IS NOT ENABLED
10822  047132  012746  177777                 MOV     #-1,-(SP)       ;ASSUME ENABLED
10823  047136  032737  000100  050456         BIT     #ECH,500$       ;ARE DATA FIELD ERRORS ENABLED ??
10824  047144  001002                         BNE     245$            ;YES !!
10825  047146  042716  000400                 BIC     #MDPE,(SP)      ;DISBALE MDPE
10826  047152  013737  001336  001142 245$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10827  047160  042637  001142                 BIC     (SP)+,$BDDAT    ;CLEAR MDPE IF ENABLED
10828  047164  001412                         BEQ     250$            ;BRANCH IF MDPE OK
10829  047166  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10830  047172  112776  000027  000000         MOVB    #27,@(SP)       ;WRITE ERROR NUMBER IN CALL
10831  047200  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10832  047204  004736                         JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10833  047206  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10834  047212                          250$:
10835
10836                                   ;REPORT ERROR IF DCK IS SET AND DATA FIELD ERRORS ARE NOT ENABLED
10837  047212  012746  177777                 MOV     #-1,-(SP)       ;ASSUME ENABLED
10838  047216  032737  000100  050456         BIT     #ECH,500$       ;ARE THEY ENABLED ??
10839  047224  001002                         BNE     255$            ;YES !!
10840  047226  042716  100000                 BIC     #DCK,(SP)       ;DISABLE DCK
10841  047232  013737  001342  001142 255$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10842  047240  042637  001142                 BIC     (SP)+,$BDDAT    ;CLEAR DCK IF ENABLED
10843  047244  001412                         BEQ     260$            ;BRANCH IF DCK IS OK
10844  047246  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10845  047252  112776  000030  000000         MOVB    #30,@(SP)       ;WRITE ERROR NUMBER IN CALL
10846  047260  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10847  047264  004736                         JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10848  047266  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10849  047272                          260$:
10850
10851                                   ;REPORT ERROR IF ECH IS SET AND,
10852                                   ;       DATA FIELD ERRORS ARE NOT ENABLED, OR
10853                                   ;       ECI IS SET, OR
10854                                   ;       DCK IS NOT SET.
10855  047272  012746  177777                 MOV     #-1,-(SP)       ;ASSUME ENABLED
10856  047276  032737  000100  050456         BIT     #ECH,500$       ;ARE ERRORS ENABLED ??
10857  047304  001410                         BEQ     265$            ;NO !!
10858  047306  032737  004000  001360         BIT     #ECI,RMOFI      ;IS ECI SET ??
10859  047314  001004                         BNE     265$            ;YES !!
10860  047316  032737  100000  001342         BIT     #DCK,RMER1I     ;IS DCK ALSO SET ??
10861  047324  001002                         BNE     270$            ;YES !!
10862  047326  042716  000100          265$:  BIC     #ECH,(SP)       ;DISABLE ECH
10863  047332  013737  001342  001142 270$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10864  047340  042637  001142                 BIC     (SP)+,$BDDAT    ;CLEAR ECH IF ENABLED
10865  047344  001412                         BEQ     275$            ;BRANCH IF ECH IS OK
10866  047346  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10867  047352  112776  000031  000000         MOVB    #31,@(SP)       ;WRITE ERROR NUMBER IN CALL
10868  047360  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10869  047364  004736                         JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10870  047366  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10871  047372                          275$:
10872
10873                                   ;;*****************************************************************
10874                                   ;PERFORM THE REMAINING ERROR CHECKS ONLY FOR DATA TRANSFER COMMANDS
10875
```

```
10876  047372  022737  000030  050464          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
10877  047400  103402                          BLO     280$            ;YES!!
10878  047402  000137  050430                  JMP     355$
10879
```

```
10880                                   ;THE FOLLOWING STATUS CHECKS ARE FOR DATA TRANSFER COMMANDS ONLY
10881                                   ;REPORT ERROR IF RMWC NOT ZERO AND TRE IS ZERO
10882  047406  013737  001330  001142  280$:   MOV     RMWCI,$BDDAT    ;WORD COUNT ZERO??
10883  047414  001421                          BEQ     285$            ;YES
10884  047416  032737  040000  001326          BIT     #TRE,RMCS1I     ;TRANSFER ERROR DETECTED??
10885  047424  001015                          BNE     285$            ;YES!!
10886  047426  062716  000004                  ADD     #4,(SP)
10887  047432  112776  000015  000000          MOVB    #15,@(SP)       ;ERROR NUMBER
10888  047440  005037  001140                  CLR     $GDDAT          ;GOOD DADA FOR TYPEOUT
10889  047444  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10890  047450  004736                          JSR     PC,@(SP)+       ;REPORT WORD COUNT NOT ZERO
10891  047452  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
10892  047456  000240                          NOP
10893                                   ;REPORT ERROR IF RMBA IS NOT CORRECT
10894  047460  013737  001330  001140  285$:   MOV     RMWCI,$GDDAT    ;NUMBER OF WORDS TRANSFERRED
10895  047466  163737  001400  001140          SUB     RMWCO,$GDDAT
10896  047474  006337  001140                  ASL     $GDDAT
10897  047500  063737  001402  001140          ADD     RMBAO,$GDDAT    ;EXPECTED BUS ADDRESS
10898  047506  032737  000010  001336          BIT     #BAI,RMCS2I     ;WAS BAI SET ??
10899  047514  001403                          BEQ     290$            ;NO !!
10900  047516  013737  001402  001140          MOV     RMBAO,$GDDAT    ;ADDRESS SHOULD NOT HAVE CHANGED
10901  047524  023737  001140  001332  290$:   CMP     $GDDAT,RMBAI    ;BUS ADDRESS OK??
10902  047532  001416                          BEQ     295$            ;YES!!
10903  047534  013737  001332  001142          MOV     RMBAI,$BDDAT    ;BAD DATA FOR TYPEOUT
10904  047542  062716  000004                  ADD     #4,(SP)
10905  047546  112776  000016  000000          MOVB    #16,@(SP)       ;ERROR NUMBER
10906  047554  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10907  047560  004736                          JSR     PC,@(SP)+       ;REPORT UNEXPECTED ADDRESS
10908  047562  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10909  047566  000240                          NOP
10910                                   ;COMPUTE NUMBER OF SECTORS TRANSFERRED
10911  047570  005046                  295$:   CLR     -(SP)           ;NUMBER OF SECTORS TRANSFERRED
10912  047572  013746  001330                  MOV     RMWCI,-(SP)     ;NUMBER OF WORDS TRANSFERRED
10913  047576  163716  001400                  SUB     RMWCO,(SP)
10914  047602  012746  000400                  MOV     #256.,-(SP)     ;NUMBER OF WORDS PER SECTOR
10915  047606  032737  000002  001376          BIT     #BIT1,RMCS10    ;HEADER & DATA COMMAND ??
10916  047614  001402                          BEQ     300$            ;NO !!
10917  047616  012716  000402                  MOV     #258.,(SP)      ;YES - CHANGE WORDS PER SECTOR
10918  047622  005266  000004          300$:   INC     4(SP)           ;INCREMENT SECTOR COUNT
10919  047626  161666  000002                  SUB     (SP),2(SP)      ;SUBTRACT ONE SECTOR'S WORTH
10920  047632  003373                          BGT     300$            ;CONTINUE IF NOT DONE
10921  047634  022626                          CMP     (SP)+,(SP)+     ;STRIP 2 FROM STACK
10922                                   ;COMPUTE EXPECTED SECTOR, TRACK AND CYLINDER ADDRESS FROM NUMBER OF SECTORS
10923  047636  013737  001404  050462          MOV     RMDAO,510$      ;STORE ORIGINAL SECTOR
10924  047644  013737  001404  050460          MOV     RMDAO,505$      ;STORE ORIGINAL TRACK
10925  047652  013737  001432  050456          MOV     RMDCO,500$      ;STORE ORIGINAL CYLINDER
10926  047660  042737  177740  050462          BIC     #^C37,510$
10927  047666  000337  050460                  SWAB    505$
10928  047672  042737  177770  050460          BIC     #^C7,505$
10929  047700  062637  050462                  ADD     (SP)+,510$
10930
10931  047704  023727  050462  000040  305$:   CMP     510$,#32.       ;SECTOR OVEFLOWED??
10932  047712  103420                          BLO     315$            ;NO!!
10933  047714  023727  050462  000240          CMP     510$,#<5*32.>   ;CYLINDERS WORTH??
10934  047722  103406                          BLO     310$            ;NO!!
10935  047724  005237  050456                  INC     500$            ;YES INCREMENT CYLINDER
```

```
10936  047730  162737  000240  050462          SUB     #(5*32.),510$   ;ADJUST SECTOR
10937  047736  000762                          BR      305$            ;TRY AGAIN
10938  047740  005237  050460          310$:   INC     505$            ;INCREMENT TRACK
10939  047744  162737  000040  050462          SUB     #32.,510$       ;ADJUST SECTOR
10940  047752  000754                          BR      305$            ;TRY AGAIN
10941
10942  047754  023727  050460  000005  315$:   CMP     505$,#5 ;TRACK OVERFLOWED??
10943  047762  103406                          BLO     320$            ;NO!!
10944  047764  005237  050456                  INC     500$            ;INCREMENT CYLINDER
10945  047770  162737  000005  050460          SUB     #5,505$ ;ADJUST TRACK
10946  047776  000766                          BR      315$            ;TRY AGAIN
10947                          ;REPORT ERROR IF "LBT" IS NOT CORRECT
10948  050000  005037  001140          320$:   CLR     $GDDAT          ;SET GOOD DATA FOR LBT=0
10949  050004  022737  001467  050456          CMP     #823.,500$      ;SHOULD LBT BE SET??
10950  050012  101007                          BHI     325$            ;NO!!
10951  050014  032737  002000  001342          BIT     #IAE,RMER1I     ;WAS IAE SET ??
10952  050022  001003                          BNE     325$            ;YES - LBT SHOULD NOT BE SET
10953  050024  012737  002000  001140          MOV     #LBT,$GDDAT     ;SET GOOD DATA FOR LBT=1
10954  050032  013737  001340  001142  325$:   MOV     RMDS1,$BDDAT    ;BAD DATA FOR TYPEOUT
10955  050040  042737  175777  001142          BIC     #^CLBT,$BDDAT
10956  050046  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS LBT CORRECT??
10957  050054  001413                          BEQ     330$            ;YES!!
10958  050056  062716  000004                  ADD     #4,(SP)
10959  050062  112776  000017  000000          MOVB    #17,@(SP)       ;ERROR NUMBER
10960  050070  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10961  050074  004736                          JSR     PC,@(SP)+       ;REPORT LBT IS WRONG
10962  050076  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10963  050102  000240                          NOP
10964                          ;REPORT ERROR IF "AOE" IS INCORRECT
10965  050104  005037  001140          330$:   CLR     $GDDAT          ;SET FOR AOE=0
10966  050110  032737  002000  001342          BIT     #IAE,RMER1I     ;WAS "IAE" DETECTED??
10967  050116  001031                          BNE     340$            ;YES-"AOE" SHOULD BE ZERO
10968  050120  022737  001467  050456          CMP     #823.,500$      ;SHOULD AOE BE SET??
10969  050126  101025                          BHI     340$            ;NO!!
10970  050130  005737  050460                  TST     505$            ;MAYBE
10971  050134  001012                          BNE     335$            ;YES
10972  050136  005737  050462                  TST     510$
10973  050142  001007                          BNE     335$            ;YES !!
10974  050144  032737  000010  050464          BIT     #F2,515$        ;WAS THIS READ OR WRITE CHECK ??
10975  050152  001413                          BEQ     340$            ;NO !!
10976  050154  005737  001330                  TST     RMWCI           ;WAS ALL DATA TRANSFERRED ??
10977  050160  001410                          BEQ     340$            ;YES !!
10978  050162  012737  001000  001140  335$:   MOV     #AOE,$GDDAT     ;SET FOR AOE=1
10979  050170  005037  050460                  CLR     505$            ;CLEAR EXPECTED TRACK
10980  050174  012737  000001  050462          MOV     #1,510$ ;EXPECT SECTOR=1
10981  050202  013737  001342  001142  340$:   MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
10982  050210  042737  176777  001142          BIC     #^CAOE,$BDDAT
10983  050216  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS AOE CORRECTY??
10984  050224  001413                          BEQ     345$            ;YES!!
10985  050226  062716  000004                  ADD     #4,(SP)
10986  050232  112776  000020  000000          MOVB    #20,@(SP)       ;ERROR NUMBER
10987  050240  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10988  050244  004736                          JSR     PC,@(SP)+       ;REPORT AOE IS WRONG
10989  050246  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10990  050252  000240                          NOP
10991                          ;REPORT ERROR IF RMDA IS NOT CORRECT
```

```
10992  050254  032737  002000  001342  345$:   BIT     #IAE,RMER1I     ;WAS THERE AN IAE ERROR ??
10993  050262  001062                          BNE     355$            ;YES - DONT CHECK RMDA,RMDC
10994  050264  013737  050460  001140          MOV     505$,$GDDAT     ;SETUP EXPECTED DISK ADDRESS
10995  050272  000337  001140                  SWAB    $GDDAT
10996  050276  113737  050462  001140          MOVB    510$,$GDDAT
10997  050304  013737  001334  001142          MOV     RMDAI,$BDDAT    ;SETUP RECEIVED DISK ADDRESS
10998  050312  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10999  050320  001413                          BEQ     350$            ;BRANCH IF EQUAL
11000  050322  062716  000004                  ADD     #4,(SP)
11001  050326  112776  000021  000000          MOVB    #21,@(SP)       ;ERROR NUMBER
11002  050334  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11003  050340  004736                          JSR     PC,@(SP)+       ;REPORT BAD DISK ADDRESS
11004  050342  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11005  050346  000240                          NOP
11006                          ;REPORT ERROR IF RMDC IS INCORRECT
11007  050350  013737  050456  001140  350$:   MOV     500$,$GDDAT     ;SETUP EXPECTED CYLINDER
11008  050356  042737  176000  001140          BIC     #^C1777,$GDDAT
11009  050364  013737  001362  001142          MOV     RMDCI,$BDDAT    ;SETUP RECEIVED CYLINDER
11010  050372  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE CYLINDERS
11011  050400  001413                          BEQ     355$            ;BRANCH IF EQUAL
11012  050402  062716  000004                  ADD     #4,(SP)
11013  050406  112776  000022  000000          MOVB    #22,@(SP)       ;ERROR NUMBER
11014  050414  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11015  050420  004736                          JSR     PC,@(SP)+       ;REPORT BAD CYLINDER
11016  050422  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11017  050426  000240                          NOP
```

```
11018  050430  062716  000004          355$:   ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
11019  050434  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
11020  050440  001403                          BEQ     360$
11021  050442  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR RETURN
11022  050446  000402                          BR      365$
11023  050450  162716  000004          360$:   SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
11024  050454  000207                  365$:   RTS     PC
11025
11026  050456  000000                  500$:   .WORD   0               ;CYLINDER
11027  050460  000000                  505$:   .WORD   0               ;TRACK
11028  050462  000000                  510$:   .WORD   0               ;SECTOR
11029  050464  000000                  515$:   .WORD   0               ;FUNCTION CODE
11030
11031
```

```
                                        .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

050466                                  DEVSEL:

                                        ;CLEAR USER'S ERROR CALL
050466  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
050472  105076  000000                          CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF CALL
050476  162716  000004                          SUB     #4,(SP)         ;MOVE SP BACK
                                        ;SAVE USER'S INFORMATION AND SETUP REGISTERS
050502  013746  000004                          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
050506  013746  000006                          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
050512  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
050514  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
050516  012737  050636  000004                  MOV     #20$,ERRVEC     ;SETUP FOR BUS TIMEOUT
050524  012737  000300  000006                  MOV     #PR6,ERRVEC+2
050532  013700  001276                          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
050536  013701  001446                          MOV     TSTQUE,R1       ;R1 POINTS TO DEVICE NUMBER

                                        ;SELECT DEVICE AND VERIFY THAT DEVICE IS AVAILABLE
050542  111160  000010                          MOVB    (R1),RMCS2(R0)  ;WRITE UNIT SELECT BITS
050546  016037  000000  001176                  MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
050554  016037  000010  001174                  MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
050562  032737  010000  001174                  BIT     #NED,$TMP0      ;IS DEVICE NONEXISTENT??
050570  001407                                  BEQ     10$             ;NO!!
050572  062766  000004  000010                  ADD     #4,10(SP)       ;MOVE SP TO USERS ERROR CALL
050600  112776  000111  000010                  MOVB    #111,@10(SP)    ;WRITE ERROR NUMBER
050606  000422                                  BR      30$             ;
050610  032737  004000  001176  10$:            BIT     #DVA,$TMP1      ;IS DEVICE AVAILABLE??
050616  001021                                  BNE     35$             ;YES!!
050620  062766  000004  000010                  ADD     #4,10(SP)
050626  112776  000112  000010                  MOVB    #112,@10(SP)
050634  000407                                  BR      30$

050636                                  20$:

                                        ;HANDLE BUS TIMEOUT
050636  022626                                  CMP     (SP)+,(SP)+     ;ADJUST SP
050640  062766  000004  000010                  ADD     #4,10(SP)
050646  112776  000113  000010                  MOVB    #113,@10(SP)
050654  162766  000002  000010  30$:            SUB     #2,10(SP)       ;MOVE SP TO RETURN IF ERROR

050662                                  35$:

                                        ;RESTORE USERS DATA AND RETURN TO ADDRESS ON STACK
050662  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
050664  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
```

```
11088  050666  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
11089  050672  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
11090  050676  000207                        RTS     PC              ;EXIT
```

```
11091                                   .SBTTL  SEEK STATUS CHECK SUBROUTINE
11092                                   ;
11093                                   ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEEK TESTS USING STATUS
11094                                   ;STORED IN THE GET BUFFER AND TEST PARAMETERS STORED IN THE PUT BUFFER.
11095                                   ;
11096                                   ;
11097                                   ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS DETECTED
11098                                   ;AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN THE "ERROR" TRAP
11099                                   ;OF THE CALLING ROUTINE.  SEEK STATUS IS CHECKED AS FOLLOWS:
11100                                   ;
11101                                   ;CALL:
11102                                   ;(1)    JSR     PC,SEKSTS
11103                                   ;       BR      ???             RETURN HERE IF NO ERROR
11104                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
11105                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11106                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11107                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
11108
11109  050700                           SEKSTS:
11110
11111                                   ;CLEAR USER'S ERROR  CALL
11112  050700  000240                          NOP
11113  050702  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
11114  050706  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
11115  050712  162716  000004                  SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
11116  050716  005037  052136                  CLR     300$            ;CLEAR ERROR FLAGS
11117
11118                                   ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING
11119                                   ;LOCAL REGISTERS, I.E., "PAR"  = 1 AND "DPE" = 0
11120  050722  032737  000010  001342          BIT     #PAR,RMER1I     ;WAS PARITY ERROR DETECTED??
11121  050730  001424                          BEQ     1$              ;NO!!
11122  050732  032737  000010  001370          BIT     #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
11123  050740  001020                          BNE     1$              ;NOT SURE!!
11124
11125                                   ;REPORT REGISTER PARITY ERROR VIA USER'S ERROR CALL
11126  050742  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
11127  050746  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11128  050754  062716  000004                  ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
11129  050760  112776  000050  000000          MOVB    #50,@(SP)       ;ERROR #50
11130  050766  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11131  050772  004736                          JSR     PC,@(SP)+
11132  050774  162716  000010                  SUB     #10,(SP)        ;RESTORE STACK
11133  051000  000437                          BR      3$              ;IAE SHOULD  BE ZERO
11134
11135
11136                                   ;DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK,SECTOR AND
11137                                   ;CYLINDER ADDRESS USED DURING SEEK OPERATION
11138  051002  012737  002000  001140  1$:     MOV     #IAE,$GDDAT     ;SET UP FOR IAE = 1
11139  051010  052737  040000  052136          BIS     #SKI,300$       ;SET SKI ERROR FLAG
11140  051016  023727  001432  001466          CMP     RMDC0,#822.     ;CYLINDER > 822??
11141  051024  101025                          BHI     3$              ;YES!!
11142  051026  042737  040000  052136          BIC     #SKI,300$       ;CLEAR SKI ERROR FLAG
11143  051034  123727  001404  000037          CMPB    RMDA0,#31.      ;SECTOR > 31??
11144  051042  101016                          BHI     3$              ;YES!!
11145  051044  123727  001404  000035          CMPB    RMDA0,#29.      ;SECTOR > 29 ??
11146  051052  101404                          BLOS    2$              ;NO!!
```

```
11147  051054  032737  010000  001430          BIT     #FMT16,RMOF0    ;30 SECTOR FORMAT??
11148  051062  001406                          BEQ     3$              ;YES!!
11149  051064  123727  001405  000004  2$:     CMPB    RMDA0+1,#4.     ;TRACK  >4??
11150  051072  101002                          BHI     3$              ;YES!!
11151  051074  005037  001140                  CLR     $GDDAT          ;"IAE" SHOULD BE 0
11152
11153                                  ;COMPARE EXPECTED AND RECEIVED "IAE" STATUS
11154  051100  013737  001342  001142  3$:     MOV     RMER1I,$BDDAT   ;IS IAE OK??
11155  051106  042737  175777  001142          BIC     #^CIAE,$BDDAT
11156  051114  023737  001140  001142          CMP     $GDDAT,$BDDAT
11157  051122  001004                          BNE     35$             ;IAE IN ERROR
11158  051124  042737  040000  052136          BIC     #SKI,300$       ;CLEAR SKI FLAG
11159  051132  000413                          BR      5$              ;GO CHECK NEXT ERROR
11160  051134                          35$:
11161                                  ;REPORT INCORRECT "IAE" STATUS VIA USER'S ERROR CALL
11162  051134  062716  000004                  ADD     #4,(SP)
11163  051140  112776  000051  000000          MOVB    #51,@(SP)       ;ERROR 51
11164  051146  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11165  051152  004736                          JSR     PC,@(SP)+       ;REPORT INCORRECT IAE
11166  051154  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11167  051160  000240                          NOP
11168  051162                          5$:
11169
11170                                  ;REPORT ANY IVC ERROR AS
11171                                  ;       IVC ERROR WITH VOLUME VALID ZERO
11172                                  ;       ERRONEOUS IVC ERROR, VOLUME VALID IS SET
11173  051162  032737  010000  001370          BIT     #IVC,RMER2I     ;IVC ERROR??
11174  051170  001427                          BEQ     52$             ;NO!!
11175  051172  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
11176  051176  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11177  051204  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
11178  051210  112776  000060  000000          MOVB    #60,@(SP)       ;ERROR 60 IF VV = 0
11179  051216  032737  000100  001340          BIT     #VV,RMDSI
11180  051224  001403                          BEQ     51$
11181  051226  112776  000061  000000          MOVB    #61,@(SP)       ;ERROR 61 IF VV  = 1
11182  051234  162716  000002          51$:    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11183  051240  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11184  051242  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
11185  051246  000240                          NOP
11186
11187  051250  013737  001370  001142  52$:     MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11188  051256  042737  137777  001142          BIC     #^CSKI,$BDDAT   ;CLEAR DONT CARES
11189  051264  013737  052136  001140          MOV     300$,$GDDAT     ;GET EXPECTED SKI STATUS
11190  051272  042737  137777  001140          BIC     #^CSKI,$GDDAT   ;CLEAR DONT CARES
11191  051300  001417                          BEQ     53$             ;BRANCH IF 0 EXPECTED
11192
11193                                  ;REPORT ERROR IF SKI IS NOT SET (IAE WAS NOT DETECTED)
11194  051302  032737  040000  001142          BIT     #SKI,$BDDAT     ;WAS SKI DETECTED ??
11195  051310  001032                          BNE     54$             ;YES !!
11196  051312  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11197  051316  112776  000267  000000          MOVB    #267,@(SP)      ;WRITE ERROR NUMBER
11198  051324  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
11199  051330  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11200  051332  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11201  051336  000443                          BR      6$              ;GO TO NEXT ERROR CHECK
11202  051340                          53$:
```

```
11203
11204                                          ;REPORT ERROR IF SKI IS SET
11205  051340  032737  040000  001142          BIT     #SKI,$BDDAT     ;IS SKI SET ??
11206  051346  001413                          BEQ     54$             ;NO - SKI IS OK
11207  051350  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR
11208  051354  112776  000054  000000          MOVB    #54,@(SP)       ;LOAD ERROR NUMBER
11209  051362  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11210  051366  004736                          JSR     PC,@(SP)+       ;REPORT SEEK ERROR
11211  051370  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
11212  051374  000240                          NOP
11213
11214                                          ;REPORT ANY DEVICE CHECK
11215  051376  032737  000200  001370  54$:    BIT     #DVC,RMER2I     ;WAS THERE DVC DURING SEEK??
11216  051404  001420                          BEQ     6$              ;NO!!
11217  051406  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
11218  051412  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11219  051420  062716  000004                  ADD     #4,(SP)
11220  051424  112776  000055  000000          MOVB    #55,@(SP)       ;ERROR  #55
11221  051432  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11222  051436  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11223  051440  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
11224  051444  000240                          NOP
11225
11226                                          ;REPORT ANY "OPI" ERROR AS OPI WITH MOL  =0, OR  OPI
11227                                          ;BECAUSE ON CYLINDER LATCH DIDN'T RESET
11228  051446  032737  020000  001342  6$:     BIT     #OPI,RMER1I     ;"OPI" ERROR??
11229  051454  001427                          BEQ     8$              ;NO!!
11230  051456  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
11231  051462  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11232  051470  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR
11233  051474  112776  000052  000000          MOVB    #52,@(SP)       ;LOAD ERROR NUMBER
11234  051502  032737  010000  001340          BIT     #MOL,RMDSI      ;WAS MEDIUM ON LINE??
11235  051510  001403                          BEQ     7$              ;NO!!
11236  051512  112776  000053  000000          MOVB    #53,@(SP)       ;YES - CHANGE ERROR NUMBER
11237  051520  162716  000002          7$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11238  051524  004736                          JSR     PC,@(SP)+       ;REPORT "OPI" ERROR
11239  051526  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
11240  051532  000240                          NOP
11241
11242                                          ;SEE IF "PIP" IS 0, AND "ATA","MOL"AND"VV" =1
11243  051534  013746  001340          8$:     MOV     RMDSI,-(SP)
11244  051540  042716  047677                  BIC     #^C<ATA!PIP!MOL!VV>,(SP)
11245  051544  022726  110100                  CMP     #ATA!MOL!VV,(SP)+
11246  051550  001002                          BNE     9$              ;ERROR IN RMDS
11247  051552  000137  052106                  JMP     14$             ;RMDS IS OK
11248
11249                                          ;REPORT ERROR IF MOL = 0 AND OPI = 0
11250  051556  032737  010000  001340  9$:     BIT     #MOL,RMDSI      ;IS MOL RESET??
11251  051564  001030                          BNE     10$             ;NO - MOL IS SET
11252  051566  032737  020000  001342          BIT     #OPI,RMER1I     ;WAS OPI SET
11253  051574  001024                          BNE     10$             ;YES - DONT REPORT ERROR
11254  051576  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11255  051604  052737  010000  001140          BIS     #MOL,$GDDAT
11256  051612  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11257  051620  062716  000004                  ADD     #4,(SP)
11258  051624  112776  000062  000000          MOVB    #62,@(SP)
```

```
11259  051632  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11260  051636  004736                    JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
11261  051640  162716  000010            SUB     #10,(SP)
11262  051644  000240                    NOP
11263
11264                                  ;REPORT AN ERROR IF "PIP" IS STIL SET AND SKI NOT SET
11265  051646  032737  020000  001340  10$:  BIT     #PIP,RMDSI       ;IS "PIP" STILL SET??
11266  051654  001430                    BEQ     11$              ;NO!!
11267  051656  032737  040000  001370        BIT     #SKI,RMER2I      ;WAS "SKI"SET??
11268  051664  001024                    BNE     11$              ;YES-DONT REPORT PIP
11269  051666  013737  001340  001140        MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
11270  051674  042737  020000  001142        BIC     #PIP,$BDDAT
11271  051702  013737  001340  001142        MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
11272  051710  062716  000004                ADD     #4,(SP)          ;MOVE (SP) TO ERROR
11273  051714  112776  000056  000000        MOVB    #56,@(SP)        ;LOAD ERROR NUMBER
11274  051722  162716  000002                SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11275  051726  004736                        JSR     PC,@(SP)+        ;REPORT "PIP" SET AFTER SEEK
11276  051730  162716  000010                SUB     #10,(SP)                 ;RESTORE (SP)
11277  051734  000240                        NOP
11278
11279                                  ;REPORT AN ERROR IF "ATA" IS NOT SET
11280  051736  032737  100000  001340  11$:  BIT     #ATA,RMDSI       ;WAS "ATA" SET ??
11281  051744  001024                    BNE     13$              ;YES!!
11282  051746  013737  001340  001140        MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
11283  051754  052737  110600  001140        BIS     #ATA!MOL!DPR!DRY,$GDDAT
11284  051762  013737  001340  001142        MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
11285  051770  062716  000004                ADD     #4,(SP)          ;MOVE (SP) TO ERROR
11286  051774  112776  000057  000000        MOVB    #57,@(SP)        ;LOAD ERROR NUMBER
11287  052002  162716  000002                SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11288  052006  004736                        JSR     PC,@(SP)+        ;REPORT ATTENTION NOT SET DURING
11289                                                                   ;SEEK TEST
11290  052010  162716  000010                SUB     #10,(SP)                 ;RESTORE (SP)
11291  052014  000240                        NOP
11292
11293                                  ;REPORT ERROR IF VOLUME VALID IS RESET AND IVC IS ZERO
11294  052016  032737  000100  001340  13$:  BIT     #VV,RMDSI        ;IS VV = 0 ??
11295  052024  001030                    BNE     14$              ;NO!!
11296  052026  032737  010000  001370        BIT     #IVC,RMER2I      ;IS IVC ALSO 0 ??
11297  052034  001024                    BNE     14$              ;NO - IVC IS SET
11298  052036  013737  001340  001140        MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
11299  052044  052737  000100  001140        BIS     #VV,$GDDAT
11300  052052  013737  001340  001142        MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
11301  052060  062716  000004                ADD     #4,(SP)
11302  052064  112776  000064  000000        MOVB    #64,@(SP)        ;ERROR #64
11303  052072  162716  000002                SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11304  052076  004736                        JSR     PC,@(SP)+
11305  052100  162716  000010                SUB     #10,(SP)
11306  052104  000240                        NOP
11307  052106                          14$:
11308
11309                                  ;MODIFY THE RETURN ADDRESS IF AN ERROR WAS DETECTED
11310  052106  000240                        NOP
11311  052110  062716  000004                ADD     #4,(SP)          ;MOVE (SP) TO ERROR CALL
11312  052114  105776  000000                TSTB    @(SP)            ;WAS ERROR CALLED??
11313  052120  001403                        BEQ     15$              ;NO!!
11314  052122  062716  000004                ADD     #4,(SP)          ;MOVE TO ERROR RETURN
```

```
 11315  052126  000402                    BR      16$
 11316
 11317  052130  162716  000004    15$:    SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
 11318  052134  000207              16$:    RTS     PC              ;RETURN
 11319
 11320  052136  000000              300$:   .WORD   0               ;ERROR FLAGS
 11321
```

```
                                        .SBTTL  CONTROLLER CLEAR SUBROUTINE

                                        ;THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTERS,
                                        ;AND DRIVES, THEN SELECTS THE DRIVE.
                                        ;
                                        ;CALL:  JSR     PC,CNTCLR
                                        ;       BR      ???             RETURN HERE IF NO ERROR FOUND
                                        ;       NOP                     RETURN HERE IF ANY ERROR FOUND
                                        ;       ERROR                   SUB DEFINES ERROR NUMBER
                                        ;       ???
                                        ;
                                        CNTCLR:
                                                MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                                                MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                                                MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
                                                MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
                                                MOV     #10$,ERRVEC     ;SETUP FOR BUS TIMEOUT
                                                MOV     #PR6,ERRVEC+2
                                                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
                                                MOV     #CLR,RMCS2(R0)  ;CLEAR MASSBUS
                                                MOV     TSTQUE,R1
                                                MOVB    (R1),RMCS2(R0)  ;SELECT DEVICE
                                                BR      20$
                                        10$:    CMP     (SP)+,(SP)+     ;ADJUST STACK
                                                ADD     #4,10(SP)       ;MOVE SP TO USER'S ERROR CALL
                                                MOVB    #7,@10(SP)      ;WRITE THE ERROR NUMBER
                                                SUB     #2,10(SP)
                                        20$:
                                                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
                                                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
                                                MOV     (SP)+,R1        ;;POP STACK INTO R1
                                                MOV     (SP)+,R0        ;;POP STACK INTO R0
                                                RTS     PC
```

```
                                .SBTTL  CONTROLLER CLEAR STATUS CHECK SUBROUTINE

                                ;THIS SUBROUTINE VERIFIES THAT THE RM03 SUBSYSTEM IS INITIALIZED BASED ON
                                ;STATUS STORED IN THE GET BUFFER.  THIS SUBROUTINE SHOULD ONLY BE
                                ;USED FOLLOWING A CONTROLLER CLEAR OPERATION, I.E., WRITING A 1 IN BIT
                                ;5 OF RMCS2, BECAUSE THE ERROR MESSAGES ARE BASED ON THAT CONDITION.

                                ;STATUS PERTINENT TO THE DEVICE IS NOT CHECKED.  IN PARTICULAR, THE
                                ;FOLLOWING STATUS BITS ARE NOT CHECKED:
                                ;
                                ;       ATA,ERR,PIP,MOL,WRL,LBT,PGM,VV,OM,UNS,SKI,DVC
                                ;

                                ;CALL:
                                ;(1)    JSR     PC,CLRSTS
                                ;       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

052256                          CLRSTS:

                                ;CLEAR USER'S ERROR CALL
                                        ADD     #4,(SP)                 ;MOVE SP TO ERROR
                                        CLRB    @(SP)                   ;CLEAR ERROR NUMBER
                                        SUB     #4,(SP)                 ;MOVE SP BACK TO NO ERROR
                                ;REPORT ERROR IF RMCS1 NOT INITIALIZED
052272  013737  001326  001142  4$:     MOV     RMCS1I,$BDDAT           ;VERIFY RMCS1
052300  042737  100000  001142          BIC     #SC,$BDDAT              ;IGNORE SPECIAL CONDITION
052306  012737  004200  001140          MOV     #DVA!RDY,$GDDAT         ;EXPECT DVA & RDY
052314  023737  001140  001142          CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED, RECEIVED
052322  001413                          BEQ     5$                      ;BRANCH IF EQUAL
052324  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
052330  112776  000126  000000          MOVB    #126,@(SP)              ;WITE ERROR NUMBER IN CALL
052336  162716  000002                  SUB     #2,(SP)                 ;MOVE SP TO RETURN FOR ERROR
052342  004736                          JSR     PC,@(SP)+               ;REPORT ERROR VIA USER
052344  162716  000010                  SUB     #10,(SP)                ;MOVE SP BACK TO NO ERROR
052350  000240                          NOP
                                ;REPORT ERROR IF RMBA NOT RESET
052352  005037  001140          5$:     CLR     $GDDAT                  ;VERIFY RMBA IS ZERO
052356  013737  001332  001142          MOV     RMBAI,$BDDAT
052364  001413                          BEQ     7$                      ;BRANCH IF ZERO
052366  062716  000004                  ADD     #4,(SP)                 ;MOVE SP TO USER'S ERROR CALL
052372  112776  000127  000000          MOVB    #127,@(SP)              ;WITE ERROR NUMBER IN CALL
052400  162716  000002                  SUB     #2,(SP)                 ;MOVE SP TO RETURN FOR ERROR
052404  004736                          JSR     PC,@(SP)+               ;REPORT ERROR VIA USER
052406  162716  000010                  SUB     #10,(SP)                ;MOVE SP BACK TO NO ERROR
052412  000240                          NOP
                                ;REPORT ERROR IF RMCS2 NOT INITIALIZED
052414  013737  001336  001142  7$:     MOV     RMCS2I,$BDDAT           ;VERIFY RMCS2
052422  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
052424  005046                          CLR     -(SP)                   ;EXPECT IR & UNIT NUMBER
052426  013701  001446                  MOV     TSTQUE,R1               ;R1 = ADDRESS OF TEST QUE
052432  111116                          MOVB    (R1),(SP)
052434  052716  000100                  BIS     #IR,(SP)
```

```
11412  052440  012637  001140              MOV     (SP)+,$GDDAT
11413  052444  012601                      MOV     (SP)+,R1        ;;POP STACK INTO R1
11414  052446  023737  001140  001142      CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11415  052454  001413                      BEQ     9$              ;BRANCH IF EQUAL
11416  052456  062716  000004              ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11417  052462  112776  000130  000000      MOVB    #130,@(SP)      ;WITE ERROR NUMBER IN CALL
11418  052470  162716  000002              SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11419  052474  004736                      JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11420  052476  162716  000010              SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11421  052502  000240                      NOP
11422                          ;REPORT ERROR IF RMER1 NOT RESET-IGNORE UNS
11423  052504  005037  001140  9$:         CLR     $GDDAT          ;VERIFY RMER1
11424  052510  013737  001342  001142      MOV     RMER1I,$BDDAT
11425  052516  042737  040000  001142      BIC     #UNS,$BDDAT     ;IGNORE UNSAFE
11426  052524  001413                      BEQ     13$             ;BRANCH IF ZERO
11427  052526  062716  000004              ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11428  052532  112776  000131  000000      MOVB    #131,@(SP)      ;WITE ERROR NUMBER IN CALL
11429  052540  162716  000002              SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11430  052544  004736                      JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11431  052546  162716  000010              SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11432  052552  000240                      NOP
11433                          ;REPORT ERROR IF RMMR1 NOT INITIALIZED-IGNORE WC,LS,LST
11434  052554  013737  001352  001142  13$:     MOV     RMMR1I,$BDDAT   ;VERIFY RMMR
11435  052562  042737  000046  001142      BIC     #WC!LS!LST,$BDDAT       ;IGNORE WORD CLOCK, SCT, TRK
11436  052570  012737  000010  001140      MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA BIT
11437  052576  023737  001140  001142      CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
11438  052604  001413                      BEQ     17$             ;BRANCH IF 0
11439  052606  062716  000004              ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11440  052612  112776  000133  000000      MOVB    #133,@(SP)      ;WITE ERROR NUMBER IN CALL
11441  052620  162716  000002              SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11442  052624  004736                      JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11443  052626  162716  000010              SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11444  052632  000240                      NOP
11445                          ;REPORT AN ERROR IF RMEC2 IS NOT RESET
11446  052634  005037  001140  17$:         CLR     $GDDAT          ;EXPECT ZEROS
11447  052640  013737  001374  001142      MOV     RMEC2I,$BDDAT   ;VERIFY RMEC2=0
11448  052646  001413                      BEQ     19$
11449  052650  062716  000004              ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11450  052654  112776  000135  000000      MOVB    #135,@(SP)      ;WITE ERROR NUMBER IN CALL
11451  052662  162716  000002              SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11452  052666  004736                      JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11453  052670  162716  000010              SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11454  052674  000240                      NOP
11455                          ;REPORT ERROR IF RMMR2 NOT INITIALIZED-IGNORE RQA,RQB
11456  052676  013737  001366  001142  19$:     MOV     RMMR2I,$BDDAT   ;VERIFY RMMR2
11457  052704  042737  140000  001142      BIC     #RQA!RQB,$BDDAT
11458  052712  012737  011777  001140      MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
11459  052720  023737  001140  001142      CMP     $GDDAT,$BDDAT
11460  052726  001413                      BEQ     21$
11461  052730  062716  000004              ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11462  052734  112776  000136  000000      MOVB    #136,@(SP)      ;WITE ERROR NUMBER IN CALL
11463  052742  162716  000002              SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11464  052746  004736                      JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11465  052750  162716  000010              SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11466  052754  000240                      NOP
11467                          ;REPORT ERROR IF RMER2 NOT RESET-IGNORE SKI,DVC
```

```
11468  052756  005037  001140            21$:    CLR     $GDDAT          ;EXPECT ALL ZEROS
11469  052762  013737  001370  001142            MOV     RMER2I,$BDDAT   ;VERIFY RMER2
11470  052770  042737  040200  001142            BIC     #SKI!DVC,$BDDAT ;IGNORE DEVICE ERRORS
11471  052776  001413                            BEQ     21S$            ;BRANCH IF OTHER BITS 0
11472  053000  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11473  053004  112776  000174  000000            MOVB    #174,@(SP)      ;WITE ERROR NUMBER IN CALL
11474  053012  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11475  053016  004736                            JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11476  053020  162716  000010                    SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11477  053024  000240                            NOP
11478                                    ;REPORT ERROR IF RMDS NOT INITIALIZED
11479  053026  013737  001340  001142    215$:   MOV     RMDSI,$BDDAT    ;TEST DRIVE STATUS REGISTER
11480  053034  042737  177177  001142            BIC     #^C<DRY!DPR>,$BDDAT
11481  053042  012737  000600  001140            MOV     #DPR!DRY,$GDDAT ;EXPECTED DRIVE STATUS
11482  053050  023737  001140  001142            CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11483  053056  001413                            BEQ     22$             ;BRANCH IF EQUAL
11484  053060  062716  000004                    ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11485  053064  112776  000134  000000            MOVB    #134,@(SP)      ;WRITE ERROR NUMBER
11486  053072  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11487  053076  004736                            JSR     PC,@(SP)+       ;REPORT ERROR TO USER
11488  053100  162716  000010                    SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11489  053104  000240                            NOP
11490  053106  062716  000004            22$:    ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11491  053112  105776  000000                    TSTB    @(SP)           ;WAS AN ERROE DETECTED??
11492  053116  001403                            BEQ     23$             ;NO!!
11493  053120  062716  000004                    ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
11494  053124  000402                            BR      24$
11495  053126  162716  000004            23$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11496  053132  000240                    24$:    NOP
11497  053134  000207                            RTS     PC
```

```
                                        .SBTTL  PACK ACKNOWLEDGE STATUS CHECK

                                        ;THIS SUBROUTINE CHECKS THE RESULTS OF A PACK ACKNOWLEDGE
                                        ;COMMAND USING THE STATUS STORED IN THE GET BUFFER.  ERRORS ARE
                                        ;REPORTED TO THE USER VIA THE USER'S ERROR CALL.

                                        ;CALL:
                                        ;(1)    JSR     PC,ACKSTS
                                        ;       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

053136                                  ACKSTS:

                                        ;CLEAR USER'S ERROR CALL
053136  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
053142  105076  000000                          CLRB    @(SP)           ;CLEAR LOW ORDER BYTE
053146  162716  000004                          SUB     #4,(SP)         ;MOVE SP BACK

                                        ;REPORT AN ERROR  IF "VV" IS 0
053152  032737  000100  001340                  BIT     #VV,RMDSI       ;IS VOLUME VALID SET??
053160  001024                                  BNE     1$              ;YES!!
053162  013737  001340  001140                  MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
053170  052737  000100  001140                  BIS     #VV,$GDDAT
053176  013737  001340  001142                  MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
053204  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
053210  112776  000155  000000                  MOVB    #155,@(SP)      ;WRITE NUMBER IN ERROR CALL
053216  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
053222  004736                                  JSR     PC,@(SP)+       ;REPORT THE ERROR
053224  162716  000010                          SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
053230  000240                                  NOP
053232                                  1$:

                                        ;REPORT AN ERROR IF "MOL" IS 0
053232  032737  010000  001340                  BIT     #MOL,RMDSI      ;IS MOL SET??
053240  001024                                  BNE     2$              ;YES!!
053242  013737  001340  001140                  MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
053250  052737  010000  001140                  BIS     #MOL,$GDDAT
053256  013737  001340  001142                  MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
053264  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
053270  112776  000041  000000                  MOVB    #41,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
053276  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
053302  004736                                  JSR     PC,@(SP)+       ;REPORT TH ERROR
053304  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO BRANCH
053310  000240                                  NOP
053312                                  2$:

                                        ;SEE IF "UNS","OPI","RMR","ILR", OR "ILF" IS SET
053312  032737  060007  001342                  BIT     #UNS!OPI!RMR!ILR!ILF,RMER1I
053320  001570                                  BEQ     7$

                                        ;REPORT AN ERROR IF "UNS" IS SET
053322  032737  040000  001342                  BIT     #UNS,RMER1I     ;WAS UNS SET??
053330  001424                                  BEQ     3$              ;NO!!
```

```
11554  053332  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11555  053340  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11556  053346  042737  040000  001140          BIC     #UNS,$GDDAT
11557  053354  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11558  053360  112776  000042  000000          MOVB    #42,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11559  053366  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11560  053372  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11561  053374  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11562  053400  000240                          NOP
11563  053402                          3$:
11564
11565                                  ;REPORT ANY OPI ERROR
11566  053402  032737  020000  001342          BIT     #OPI,RMER1I     ;WAS OPI SET ??
11567  053410  001424                          BEQ     4$              ;NO!!
11568  053412  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11569  053420  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11570  053426  042737  020000  001140          BIC     #OPI,$GDDAT
11571  053434  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11572  053440  112776  000043  000000          MOVB    #43,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11573  053446  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11574  053452  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11575  053454  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11576  053460  000240                          NOP
11577  053462                          4$:
11578
11579                                  ;REPORT ANY RMR ERROR
11580  053462  032737  000004  001342          BIT     #RMR,RMER1I     ;WAS RMR SET??
11581  053470  001424                          BEQ     5$              ;NO!!
11582  053472  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11583  053500  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11584  053506  042737  000004  001140          BIC     #RMR,$GDDAT
11585  053514  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11586  053520  112776  000044  000000          MOVB    #44,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11587  053526  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11588  053532  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11589  053534  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11590  053540  000240                          NOP
11591  053542                          5$:
11592
11593                                  ;REPORT ANY     ILR ERROR
11594  053542  032737  000002  001342          BIT     #ILR,RMER1I     ;WAS ILR SET??
11595  053550  001424                          BEQ     6$              ;NO!!
11596  053552  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11597  053560  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11598  053566  042737  000002  001140          BIC     #ILR,$GDDAT
11599  053574  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11600  053600  112776  000045  000000          MOVB    #45,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11601  053606  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11602  053612  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11603  053614  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11604  053620  000240                          NOP
11605  053622                          6$:
11606
11607                                  ;REPORT ANY     ILF ERROR
11608  053622  032737  000001  001342          BIT     #ILF,RMER1I     ;WAS ILF SET??
11609  053630  001424                          BEQ     7$              ;NO!!
```

```
11610  053632  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11611  053640  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11612  053646  042737  000001  001140          BIC     #ILF,$GDDAT
11613  053654  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11614  053660  112776  000046  000000          MOVB    #46,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11615  053666  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11616  053672  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11617  053674  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11618  053700  000240                          NOP
11619  053702                          7$:
11620
11621                                  ;AUGMENT RETURN ADDRESS IF ERROR WAS FOUND
11622  053702  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11623  053706  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
11624  053712  001403                          BEQ     8$              ;NO!!
11625  053714  062716  000004                  ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
11626  053720  000402                          BR      9$
11627  053722  162716  000004          8$:     SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11628  053726  000240                  9$:     NOP
11629  053730  000207                          RTS     PC
11630
```

```
                                .SBTTL  RECALIBRATE STATUS CHECK SUBROUTINE

                                ;THIS SUBROUTINE CHECKS THE RESULTS OF A RECALIBRATE OPERATION
                                ;USING THE STATUS STORED IN THE GET BUFFER.

                                ;CALL:
                                ;
                                ;(1)    JSR     PC,RCLSTS       ;CALL SUBROUTINE
                                ;       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

053732                          RCLSTS:

                                ;CLEAR USER'S ERROR NUMBER
053732  062716  000004                ADD     #4,(SP)
053736  105076  000000                CLRB    @(SP)           ;CLEAR USER'S ERROR CALL
053742  162716  000004                SUB     #4,(SP)         ;MOVE SP BACK TO BRANCH

                                ;SEE IF "PAR" OR "ILF" OR "OPI" OR "IAE" IS SET
053746  032737  022011  001342        BIT     #OPI!PAR!ILF!IAE,RMER1I
053754  001553                      BEQ     4$              ;NONE ARE SET - GO TO NEXT CHECK

                                ;REPORT ANY MASSBUS CONTROL BUS PARITY ERROR, I.E.,
                                ;"PAR" = 1 AND "DPE" = 0
053756  032737  000010  001342        BIT     #PAR,RMER1I     ;WAS "PAR" SET??
053764  001430                      BEQ     1$              ;NO!!
053766  032737  000010  001370        BIT     #DPE,RMER2I     ;WAS "DPE" SET??
053774  001024                      BNE     1$              ;YES - NOT A REGISTER ERROR
053776  013737  001342  001140        MOV RMER1I,$GDDAT       ;EXPECTED STATUS
054004  042737  000010  001140        BIC     #PAR,$GDDAT
054012  013737  001342  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
054020  062716  000004                ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
054024  112776  000050  000000        MOVB    #50,@(SP)       ;WRITE ERROR NUMBER IN CALL
054032  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
054036  004736                      JSR     PC,@(SP)+       ;GO REPORT ERROR
054040  162716  000010                SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
054044  000240                      NOP
054046                          1$:

                                ;REPORT ANY "ILF" ERROR
054046  032737  000001  001342        BIT     #ILF,RMER1I     ;WAS "ILF" SET??
054054  001424                      BEQ     2$              ;NO!!
054056  013737  001342  001140        MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
054064  042737  000001  001140        BIC     #ILF,$GDDAT
054072  013737  001342  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
054100  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
054104  112776  000071  000000        MOVB    #71,@(SP)       ;WRITE ERROR NUMBER IN CALL
054112  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
054116  004736                      JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
054120  162716  000010                SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
054124  000240                      NOP
054126                          2$:
```

```
11687
11688                                   ;REPORT ANY "OPI" ERROR AS
11689                                   ;       . OPI DUE TO "MOL" = 0
11690                                   ;       . OPI BECAUSE ON CYLINDER LATCH DIDN'T RESET
11691  054126  032737  020000  001342           BIT     #OPI,RMER1I     ;WAS OPI SET??
11692  054134  001433                           BEQ     31$             ;NO!!
11693  054136  013737  001342  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11694  054144  042737  020000  001140           BIC     #OPI,$GDDAT
11695  054152  013737  001342  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11696  054160  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11697  054164  112776  000072  000000           MOVB    #72,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11698  054172  032737  010000  001340           BIT     #MOL,RMDSI      ;WAS "MOL" = 0??
11699  054200  001403                           BEQ     3$              ;YES!!
11700  054202  112776  000073  000000           MOVB    #73,@(SP)       ;NO - CHANGE ERROR NUMBER
11701  054210  162716  000002           3$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11702  054214  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11703  054216  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11704  054222  000240                           NOP
11705  054224                           31$:
11706
11707                                   ;REPORT AN ERROR IF "IAE" IS SET
11708  054224  032737  002000  001342           BIT     #IAE,RMER1I     ;IS "IAE" SET??
11709  054232  001424                           BEQ     4$              ;NO!!
11710  054234  013737  001342  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11711  054242  042737  002000  001140           BIC     #IAE,$GDDAT
11712  054250  013737  001342  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11713  054256  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11714  054262  112776  000070  000000           MOVB    #70,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11715  054270  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11716  054274  004736                           JSR     PC,@(SP)+       ;REPORT ERROR
11717  054276  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
11718  054302  000240                           NOP
11719  054304                           4$:
11720
11721                                   ;SEE IF "SKI" OR "IVC" OR "DVC" IS SET
11722  054304  032737  050200  001370           BIT     #SKI!IVC!DVC,RMER2I
11723  054312  001517                           BEQ     8$              ;NONE OF THE BITS ARE SET
11724
11725
11726                                   ;REPORT ANY "IVC" ERROR AS
11727                                   ;       .IVC WITH VV = 0
11728                                   ;       .ERRONEOUS IVC ERROR
11729  054314  032737  010000  001370           BIT     #IVC,RMER2I     ;WAS IVC SET??
11730  054322  001433                           BEQ     6$              ;NO!!
11731  054324  013737  001370  001140           MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11732  054332  042737  010000  001140           BIC     #IVC,$GDDAT
11733  054340  013737  001370  001142           MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11734  054346  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11735  054352  112776  000074  000000           MOVB    #74,@(SP)       ;;WRITE ERROR NUMBER IN CALL
11736  054360  032737  000100  001340           BIT     #VV,RMDSI       ;WAS VV = 0??
11737  054366  001403                           BEQ     5$              ;YES!!
11738  054370  112776  000075  000000           MOVB    #75,@(SP)       ;NO - CHANGE ERROR NUMBER
11739  054376  162716  000002           5$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11740  054402  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11741  054404  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11742  054410  000240                           NOP
```

```
11743  054412                                  6$:
11744
11745
11746  054412  032737  040000  001370          ;REPORT ANY "SKI" ERROR
11747  054420  001424                          BIT     #SKI,RMER2I     ;WAS SKI SET??
11748  054422  013737  001370  001140          BEQ     7$              ;NO!!
11749  054430  042737  040000  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11750  054436  013737  001370  001142          BIC     #SKI,$GDDAT
11751  054444  062716  000004                  MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11752  054450  112776  000076  000000          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11753  054456  162716  000002                  MOVB    #76,@(SP)       ;WRITE ERROR NUMBER
11754  054462  004736                          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11755  054464  162716  000010                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11756  054470  000240                          SUB     #10,(SP)        ;MOVE SP TO BRANCH
11757  054472                                  NOP
11758                                  7$:
11759
11760  054472  032737  000200  001370          ;REPORT ANY "DVC" ERROR
11761  054500  001424                          BIT     #DVC,RMER2I     ;WAS "DVC" SET??
11762  054502  013737  001370  001140          BEQ     8$              ;NO!!
11763  054510  042737  000200  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11764  054516  013737  001370  001142          BIC     #DVC,$GDDAT
11765  054524  062716  000004                  MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11766  054530  112776  000077  000000          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11767  054536  162716  000002                  MOVB    #77,@(SP)       ;WRITE ERROR NUMBER
11768  054542  004736                          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11769  054544  162716  000010                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11770  054550  000240                          SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11771  054552                                  NOP
11772                                  8$:
11773
11774  054552  013746  001340                  ;SEE IF "PIP" AND "OM" ARE 0, AND "ATA","MOL" AND "VV" ARE 1
11775  054556  042716  047676                  MOV     RMDSI,-(SP)     ;PUT RMDS ON STACK
11776  054562  022726  110100                  BIC     #^C<PIP!MOL!VV!OM!ATA>,(SP)
11777  054566  001002                          CMP     #ATA!MOL!VV,(SP)+
11778  054570  000137  055204                  BNE     85$
11779  054574                                  JMP     13$
11780                                  85$:
11781
11782                                  ;REPORT AN ERROR IF MOL = 0 AND OPI = 0, I.E., MEDIUM WENT OFF
11783  054574  032737  010000  001340          ;LINE AFTER RECALIBRATE WAS INITIATED
11784  054602  001030                          BIT     #MOL,RMDSI      ;DID MOL DROP??
11785  054604  032737  020000  001342          BNE     9$              ;NO!!
11786  054612  001024                          BIT     #OPI,RMER1I     ;WAS OPI ERROR REPORTED??
11787  054614  013737  001340  001140          BNE     9$              ;YES - DON'T REPORT MOL=0
11788  054622  052737  010000  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11789  054630  013737  001340  001142          BIS     #MOL,$GDDAT
11790  054636  062716  000004                  MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11791  054642  112776  000100  000000          ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11792  054650  162716  000002                  MOVB    #100,@(SP)      ;WRITE ERROR NUMBER
11793  054654  004736                          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11794  054656  162716  000010                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11795  054662  000240                          SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
11796  054664                                  NOP
11797                                  9$:
11798
                                        ;REPORT AN ERROR IF "VV" = 0 AND "IVC" = 0
```

```
11799  054664  032737  000100  001340          BIT     #VV,RMDSI       ;DID "VV" DROP??
11800  054672  001030                          BNE     10$             ;NO!!
11801  054674  032737  010000  001370          BIT     #IVC,RMER2I     ;WAS THERE A IVC ERROR??
11802  054702  001024                          BNE     10$             ;YES - DONT REPORT VV=0
11803  054704  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11804  054712  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11805  054720  052737  000100  001140          BIS     #VV,$GDDAT
11806  054726  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11807  054732  112776  000101  000000          MOVB    #1Ø1,@(SP)      ;WRITE ERROR NUMBER IN CALL
11808  054740  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11809  054744  004736                          JSR     PC,@(SP)+
11810  054746  162716  000010                  SUB     #1Ø,(SP)        ;MOVE SP BACK TO USER'S BRANCH
11811  054752  000240                          NOP
11812  054754                          10$:
11813
11814                           ;REPORT AN ERROR IF ATA IS NOT SET
11815  054754  032737  100000  001340          BIT     #ATA,RMDSI      ;WAS ATA SET DURING RECALIBRATE??
11816  054762  001024                          BNE     11$             ;YES!!
11817  054764  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11818  054772  052737  100000  001140          BIS     #ATA,$GDDAT
11819  055000  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11820  055006  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11821  055012  112776  000102  000000          MOVB    #1Ø2,@(SP)      ;WRITE ERROR NUMBER IN CALL
11822  055020  162716  000002                  SUB     #2,(SP)
11823  055024  004736                          JSR     PC,@(SP)+
11824  055026  162716  000010                  SUB     #1Ø,(SP)        ;MOVE SP TO USER'S BRANCH
11825  055032  000240                          NOP
11826
11827  055034                          11$:
11828
11829                           ;REPORT AN ERROR IF "OM" IS NOT ZERO BECAUSE RECALIBRATE SHOULD
11830                           ;ALWAYS CLEAR OFFSET MODE
11831  055034  032737  000001  001340          BIT     #OM,RMDSI       ;WAS "OM" RESET??
11832  055042  001424                          BEQ     12$             ;YES!!
11833  055044  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11834  055052  042737  000001  001140          BIC     #OM,$GDDAT
11835  055060  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11836  055066  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11837  055072  112776  000103  000000          MOVB    #1Ø3,@(SP)      ;WRITE ERROR NUMBER
11838  055100  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11839  055104  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11840  055106  162716  000010                  SUB     #1Ø,(SP)        ;MOVE SP TO USER'S BRANCH
11841  055112  000240                          NOP
11842  055114                          12$:
11843
11844                           ;REPORT AN ERROR IF "PIP" IS STIL ON, I.E., DRIVE NOT ON
11845                           ;CYLINDER
11846  055114  032737  020000  001340          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
11847  055122  001430                          BEQ     13$             ;NO!!
11848  055124  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI" DETECTED??
11849  055132  001024                          BNE     13$             ;YES-DONT REPORT "PIP"
11850  055134  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11851  055142  042737  020000  001140          BIC     #PIP,$GDDAT
11852  055150  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11853  055156  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11854  055162  112776  000104  000000          MOVB    #1Ø4,@(SP)      ;WRITE ERROR NUMBER
```

```
11855  055170  162716  000002            SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11856  055174  004736                    JSR     PC,@(SP)+
11857  055176  162716  000010            SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
11858  055202  000240                    NOP
11859  055204                          13$:
11860
11861                                   ;SEE IF "ILR" OR "RMR" OR "UNS" IS SET
11862  055204  032737  040006  001342    BIT     #ILR!RMR!UNS,RMER1I
11863  055212  001514                    BEQ     16$
11864
11865                                   ;REPORT AN ERROR IF "ILR" IS SET
11866  055214  032737  000002  001342    BIT     #ILR,RMER1I     ;WAS ILR SET DURING RECALIBRATE??
11867  055222  001424                    BEQ     14$             ;NO!!
11868  055224  013737  001342  001140    MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11869  055232  042737  000002  001140    BIC     #ILR,$GDDAT
11870  055240  013737  001342  001142    MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11871  055246  062716  000004            ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11872  055252  112776  000105  000000    MOVB    #105,@(SP)      ;WRITE ERROR NUMBER IN CALL
11873  055260  162716  000002            SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11874  055264  004736                    JSR     PC,@(SP)+
11875  055266  162716  000010            SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11876  055272  000240                    NOP
11877  055274                          14$:
11878
11879                                   ;REPORT AN ERROR IF "RMR" IS SET
11880  055274  032737  000004  001342    BIT     #RMR,RMER1I     ;WAS RMR SET??
11881  055302  001424                    BEQ     15$             ;NO!!
11882  055304  013737  001342  001140    MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11883  055312  042737  000004  001140    BIC     #RMR,$GDDAT
11884  055320  013737  001342  001142    MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11885  055326  062716  000004            ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11886  055332  112776  000106  000000    MOVB    #106,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
11887  055340  162716  000002            SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11888  055344  004736                    JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11889  055346  162716  000010            SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11890  055352  000240                    NOP
11891  055354                          15$:
11892
11893                                   ;REPORT AN ERROR IF "UNS" IS SET AND "DVC" IS 0
11894  055354  032737  040000  001342    BIT     #UNS,RMER1I     ;WAS UNSAFE ON??
11895  055362  001430                    BEQ     16$             ;NO!!
11896  055364  032737  000200  001370    BIT     #DVC,RMER2I     ;WAS THERE A DEVICE CHECK??
11897  055372  001024                    BNE     16$             ;YES - DON'T REPORT UNSAFE
11898  055374  013737  001342  001140    MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11899  055402  042737  040000  001140    BIC     #UNS,$GDDAT
11900  055410  013737  001342  001142    MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11901  055416  062716  000004            ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11902  055422  112776  000107  000000    MOVB    #107,@(SP)      ;WRITE ERROR NUMBER
11903  055430  162716  000002            SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11904  055434  004736                    JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11905  055436  162716  000010            SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
11906  055442  000240                    NOP
11907  055444                          16$:
11908
11909                                   ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
11910  055444  062716  000004            ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
```

```
11911  055450  105776  000000          TSTB    @(SP)           ;WAS AN ERROR REPORTED??
11912  055454  001403                  BEQ     17$             ;NO!!
11913  055456  062716  000004          ADD     #4,(SP)         ;YES - AUGMENT SP RETURN
11914  055462  000402                  BR      18$
11915  055464  162716  000004  17$:     SUB     #4,(SP) ;NO ERROR - RETURN SP TO BRANCH
11916  055470  000240          18$:     NOP
11917  055472  000207                  RTS     PC              ;STATUS CECK IS COMPLETE
11918
```

```
11919                                   .SBTTL  DRIVE CLEAR STATUS CHECK SUBROUTINE
11920                                   ;       BR      ???             RETURN HERE IF NO ERROR
11921                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
11922                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11923                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11924                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
11925
11926   055474                          DRVSTS:
11927
11928                                   ;CLEAR USER'S ERROR CALL
11929   055474  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11930   055500  105076  000000                        CLRB    @(SP)           ;CLEAR ERROR CALL
11931   055504  162716  000004                        SUB     #4,(SP)         ;MOVE SP TO USER'S BRANCH
11932                                   ;REPORT ERROR IF RMCS1 NOT INITIALIZED
11933   055510  013737  001326  001142 4$:     MOV     RMCS1I,$BDDAT   ;CHECK RMCS1
11934   055516  042737  173700  001142         BIC     #^C<DVA!FNCMSK>,$BDDAT  ;CLEAR DONT CARES
11935   055524  012737  004010  001140         MOV     #DVA!DRVCLR,$GDDAT      ;EXPECT DVA
11936   055532  023737  001140  001142         CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11937   055540  001443                        BEQ     6$              ;BRANCH IF EQUAL
11938   055542  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11939   055546  112776  000141  000000        MOVB    #141,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11940   055554  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11941   055560  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11942   055562  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11943   055566  000240                        NOP
11944                                   ;REPORT ERROR IF RMDS NOT INITIALIZED
11945   055570  013737  001340  001142 5$:     MOV     RMDSI,$BDDAT    ;CHECK RMDS
11946   055576  042737  021101  001142         BIC     #PGM!OM!VV!PIP,$BDDAT   ;CLEAR DONT CARES
11947   055604  012737  010600  001140         MOV     #MOL!DPR!DRY,$GDDAT     ;EXPECT DRY & DPR
11948   055612  023737  001140  001142         CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11949   055620  001413                        BEQ     6$              ;BRANCH IF EQUAL
11950   055622  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11951   055626  112776  000142  000000        MOVB    #142,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11952   055634  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11953   055640  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11954   055642  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11955   055646  000240                        NOP
11956                                   ;REPORT ERROR IF RMER1 NOT INITIALIZED
11957   055650  005037  001140         6$:     CLR     $GDDAT          ;EXPECT 0'S
11958   055654  013737  001342  001142         MOV     RMER1I,$BDDAT   ;CHECK RMER1
11959   055662  001413                        BEQ     8$              ;BRANCH IF EQUAL
11960   055664  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11961   055670  112776  000143  000000        MOVB    #143,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11962   055676  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11963   055702  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11964   055704  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11965   055710  000240                        NOP
11966                                   ;REPORT ERROR IF ATA NOT INITIALIZED
11967   055712  013737  001344  001142 8$:     MOV     RMASI,$BDDAT    ;CHECK ATTENTION BIT
11968   055720  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11969   055722  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11970   055724  013701  001446                MOV     TSTQUE,R1
11971   055730  116102  000001                MOVB    1(R1),R2
11972   055734  042702  177400                BIC     #^CATNMSK,R2
11973   055740  005102                        COM     R2
11974   055742  040237  001142                BIC     R2,$BDDAT
```

```
11975  055746  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
11976  055750  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
11977  055752  005737  001142                TST     $BDDAT          ;IS ATTENTION CLEARED??
11978  055756  001413                        BEQ     9$              ;BRANCH IF ATTENTION CLEARED
11979  055760  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11980  055764  112776  000144  000000        MOVB    #144,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11981  055772  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11982  055776  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11983  056000  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11984  056004  000240                        NOP
11985                                ;REPORT ERROR IF RMMR1 NOT INITIALIZED
11986  056006  013737  001352  001142  9$:   MOV     RMMR1I,$BDDAT   ;CHECK RMMR
11987  056014  042737  000046  001142        BIC     #WC!LS!LST,$BDDAT       ;CLEAR DONT CARES
11988  056022  012737  000010  001140        MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA ON
11989  056030  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
11990  056036  001413                        BEQ     11$             ;BRANCH IF ZERO
11991  056040  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11992  056044  112776  000145  000000        MOVB    #145,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11993  056052  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11994  056056  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11995  056060  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11996  056064  000240                        NOP
11997                                ;REPORT ERROR IF RMMR2 NOT INITIALIZED
11998  056066  013737  001366  001142  11$:   MOV     RMMR2I,$BDDAT   ;CHECK RMMR2
11999  056074  042737  140000  001142        BIC     #RQA!RQB,$BDDAT ;CLEAR REQA, REQB
12000  056102  012737  011777  001140        MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
12001  056110  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
12002  056116  001413                        BEQ     15$             ;BRANCH IF EQUAL
12003  056120  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12004  056124  112776  000146  000000        MOVB    #146,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12005  056132  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12006  056136  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12007  056140  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12008  056144  000240                        NOP
12009  056146  005037  001140        15$:    CLR     $GDDAT          ;EXPECT ZEROS
12010                                ;REPORT ERROR IF RMEC2 NOT RESET
12011  056152  013737  001374  001142        MOV     RMEC2I,$BDDAT   ;CHECK RMEC2
12012  056160  001413                        BEQ     17$             ;BRANCH IF 0
12013  056162  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12014  056166  112776  000150  000000        MOVB    #150,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12015  056174  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12016  056200  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12017  056202  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12018  056206  000240                        NOP
12019                                ;REPORT ERROR IF RMER2 NOT RESET
12020  056210  013737  001370  001142  17$:   MOV     RMER2I,$BDDAT   ;CHECK RMER2
12021  056216  001413                        BEQ     18$             ;BRANCH IF NO ERROR
12022  056220  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12023  056224  112776  000147  000000        MOVB    #147,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12024  056232  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12025  056236  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12026  056240  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12027  056244  000240                        NOP
12028  056246                        18$:
12029
12030  056246                        19$:
```

```
12031
12032                                   ;AUGMENT RETURN ADDRESS IF ANY ERROR WAS FOUND
12033  056246  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12034  056252  105776  000000                 TSTB    @(SP)           ;WAS AN ERROR DETECTED??
12035  056256  001403                          BEQ     21$             ;NO!!
12036  056260  062716  000004                 ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
12037  056264  000402                          BR      23$
12038  056266  162716  000004         21$:    SUB     #4,(SP) ;MOVE SP BACK TO NO ERROR RETURN
12039  056272  000240                 23$:    NOP
12040  056274  000207                          RTS     PC              ;RETURN TO USER
```

```
12041                                   .SBTTL  DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE
12042
12043                                   ;THIS SUBROUTINE VERIFIES THE RESULTS OF ALL DATA TRANSFER COMMANDS
12044                                   ;USING STATUS STORED IN THE GET BUFFER AND TEST PARAMETERS
12045                                   ;STORED IN THE PUT BUFFER.  ERRORS ARE REPORTED BY WRITING
12046                                   ;THE ERROR NUMBER IN THE USERS ERROR CALL.
12047
12048                                   ;USER'S SUBROUTINE CALL:
12049                                   ;(1)    JSR     PC,DTASTS
12050                                   ;(2)    BR      ??              RETURN HERE IF NO DATA ERRORS
12051                                   ;(3)    NOP                     RETURN HERE TO REPORT AN ERROR
12052                                   ;(4)    ERROR                   SUB WRITES ERROR NUMBER
12053                                   ;(5)    JSR     PC,@(SP)+       USER RETURNS FOR MORE CHECKS
12054                                   ;(6)    ??                      SUB RETURNS HERE AFTER ALL
12055                                   ;                               ERRORS ARE REPORTED
12056
12057  056276                           DTASTS:
12058
12059                                   ;CLEAR USER'S ERROR CALL AND ERROR FLAGS
12060  056276  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12061  056302  105076  000000                  CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF TRAP
12062  056306  162716  000004                  SUB     #4,(SP)         ;RESTORE SP TO NO ERROR
12063  056312  005037  061672                  CLR     500$            ;CLEAR ERROR FLAGS
12064
12065                                   ;REPORT ANY CONTROL BUS PARITY ERROR WHILE READING REMOTE REGISTERS, I.E., MCPE = 1
12066  056316  032737  020000  001326          BIT     #MCPE,RMCS1I    ;WAS THERE A PARITY ERROR??
12067  056324  001422                          BEQ     10$             ;NO!!
12068  056326  013737  001326  001140          MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
12069  056334  042737  020000  001140          BIC     #MCPE,$GDDAT
12070  056342  013737  001326  001142          MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
12071  056350  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12072  056354  112776  000013  000000          MOVB    #13,@(SP)       ;WRITE ERROR NUMBER
12073  056362  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12074  056366  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12075  056370  000466                          BR      30$
12076  056372                           10$:
12077
12078                                   ;REPORT ANY CONTROL BUS PARITY ERROR WHILE WRING REMOTE REGISTERS, I.E.,
12079                                   ;PAR=1  AND DPE = 0
12080  056372  032737  000010  001342          BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
12081  056400  001435                          BEQ     20$             ;NO!!
12082  056402  032737  000010  001370          BIT     #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
12083  056410  001031                          BNE     20$             ;NO!!
12084  056412  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12085  056420  042737  000010  001140          BIC     #PAR,$GDDAT
12086  056426  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12087  056434  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12088  056440  112776  000050  000000          MOVB    #50,@(SP)       ;WRITE ERROR NUMBER
12089  056446  032737  001000  001326          BIT     #MXF,RMCS1I     ;DID MXF GET SET??
12090  056454  001003                          BNE     15$             ;YES!!
12091  056456  112776  000274  000000          MOVB    #274,@(SP)      ;NO - CHANGE ERROR NUMBER
12092  056464  162716  000002           15$:   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12093  056470  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12094  056472  000425                          BR      30$
12095
12096  056474                           20$:
```

```
12097
12098                                   ;LOOK FOR ANY ERRORS WHICH MAY HAVE OCCURRED DURING COMMAND INITIATION OR
12099                                   ;MECHANICAL POSITIONING
12100
12101                                   ;FIRST TEST MXF WHICH WOULD INDICATE COMPOSITE ERROR SET WHEN FUNCTION
12102                                   ;CODE AND GO BIT WERE LOADED
12103  056474  032737  001000  001336           BIT     #MXF,RMCS2I     ;WAS "MISSED TRANSFER" SET??
12104  056502  001425                           BEQ     40$             ;NO!!
12105  056504  013737  001336  001140           MOV     RMCS2I,$GDDAT   ;EXPECTED STATUS
12106  056512  042737  001000  001140           BIC     #MXF,$GDDAT
12107  056520  013737  001336  001142           MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
12108  056526  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12109  056532  112776  000275  000000           MOVB    #275,@(SP)      ;WRITE ERROR NUMBER
12110  056540  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12111  056544  004736                           JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12112  056546                          30$:
12113
12114                                   ;RESTORE SP TO NO ERROR RETURN AND BYPASS FURHTER STATUS CHECKING
12115  056546  162716  000010                   SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12116  056552  000137  061644                   JMP     380$            ;SKIP TO END OF SUB
12117
12118  056556                          40$:
12119
12120                                   ;REPORT AN ERROR IF "OPI" ERROR OCCURRED DUE TO "MOL" = 0, OR IF "OPI"
12121                                   ;AND "MOL" ARE SET, BUT "VV" IS RESET, INDICATING AN INTERMITTENT
12122                                   ;"MOL"
12123  056556  032737  020000  001342           BIT     #OPI,RMER1I     ;IS "OPI" SET??
12124  056564  001447                           BEQ     60$             ;NO!!
12125  056566  013737  001342  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12126  056574  042737  020000  001140           BIC     #OPI,$GDDAT
12127  056602  013737  001342  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12128  056610  032737  010000  001340           BIT     #MOL,RMDSI      ;WAS MEDIUM OFF LINE??
12129  056616  001404                           BEQ     45$             ;YES!!
12130  056620  032737  000100  001340           BIT     #VV,RMDSI       ;WAS "MOL" INTERMITTENT??
12131  056626  001013                           BNE     50$             ;NO!!
12132  056630  062716  000004          45$:     ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12133  056634  112776  000276  000000           MOVB    #276,@(SP)      ;WRITE ERROR NUMBER IN CALL
12134  056642  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12135  056646  004736                           JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12136  056650  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12137  056654  000413                           BR      60$
12138  056656                          50$:
12139
12140                                   ;REPORT "OPI" ERROR, WHICH IS DUE TO "ON CYLINDER" NOT DROPPING OR
12141                                   ;"RUN" TIMEOUT (20 MS) OR SEARCH TIMEOUT (50 MS)
12142  056656  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12143  056662  112776  000277  000000           MOVB    #277,@(SP)      ;WRITE ERROR NUMBER IN CALL
12144  056670  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12145  056674  004736                           JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12146  056676  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12147  056702  000240                           NOP
12148  056704                          60$:
12149
12150                                   ;LOOK FOR "IVC" ERROR DURING COMMAND INITIATION
12151  056704  032737  010000  001370           BIT     #IVC,RMER2I     ;WAS THERE AN "IVC" ERROR??
12152  056712  001432                           BEQ     70$             ;NO!!
```

```
12153
12154  056714  013737  001370  001140  ;REPORT "IVC" ERROR DUE TO "VV" = 0, OR REPORT ERRONEOUS "IVC" ERROR
12155  056722  042737  010000  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12156  056730  013737  001370  001142          BIC     #IVC,$GDDAT
12157  056736  062716  000004                  MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12158  056742  112776  000300  000000          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12159  056750  032737  000100  001340          MOVB    #300,@(SP)      ;WRITE ERROR NUMBER IN CALL
12160  056756  001403                          BIT     #VV,RMDSI       ;WAS VOLUME VALID??
12161  056760  112776  000301  000000          BEQ     65$             ;NO!!
12162  056766  162716  000002                  MOVB    #301,@(SP)      ;CHANGE ERROR NUMBER
12163  056772  004736                  65$:    SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12164  056774  162716  000010                  JSR     PC,@(SP)+       ;REPORT "IVC" ERROR AND RETURN
12165  057000                          70$:    SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12166
12167                                  ;SEE IF "ILF" OR "RMR" IS SET
12168  057000  032737  000007  001342          BIT     #ILR!ILF!RMR, RMER1I
12169  057006  001510                          BEQ     100$            ;NO ERRORS DETECTED
12170                                  ;REPORT AN ERROR IF "ILR" IS SET
12171  057010  032737  000002  001342          BIT     #ILR,RMER1I     ;WAS "ILR" DETECTED??
12172  057016  001424                          BEQ     80$             ;NO!!
12173  057020  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12174  057026  042737  000002  001140          BIC     #ILR,$GDDAT
12175  057034  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12176  057042  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12177  057046  112776  000302  000000          MOVB    #302,@(SP)      ;WRITE ERROR NUMBER IN CALL
12178  057054  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12179  057060  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12180  057062  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12181  057066  000240                          NOP
12182  057070                          80$:
12183
12184                                  ;REPORT AN ERROR IF "ILF" IS SET
12185  057070  032737  000001  001342          BIT     #ILF,RMER1I     ;WAS "ILF" DETECTED??
12186  057076  001424                          BEQ     90$             ;NO!!
12187  057100  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12188  057106  042737  000001  001140          BIC     #ILF,$GDDAT
12189  057114  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12190  057122  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12191  057126  112776  000303  000000          MOVB    #303,@(SP)      ;WRITE ERROR NUMBER IN CALL
12192  057134  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12193  057140  004736                          JSR     PC,@(SP)+       ;REPORT ERORR AND RETURN
12194  057142  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12195  057146  000240                          NOP
12196  057150                          90$:
12197
12198                                  ;REPORT AN ERROR IF "RMR" IS SET
12199  057150  032737  000004  001342          BIT     #RMR,RMER1I     ;WAS "RMR" DETECTED??
12200  057156  001424                          BEQ     100$            ;NO!!
12201  057160  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12202  057166  042737  000004  001140          BIC     #RMR,$GDDAT
12203  057174  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12204  057202  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12205  057206  112776  000304  000000          MOVB    #304,@(SP)      ;WRITE ERROR NUMBER IN CALL
12206  057214  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12207  057220  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12208  057222  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
```

```
12209  057226  000240                          NOP
12210  057230                          100$:
12211                                  ;DETERMINE WHETHER OR NOT "IAE" SHOULD BE SET AND CHECK FOR ERROR
12212  057230  012737  002000  001140          MOV     #IAE,$GDDAT     ;SETUP FOR "IAE" = 1
12213  057236  052737  040000  061672          BIS     #SKI,500$       ;SET SKI FLAG
12214  057244  022737  001466  001432          CMP     #822.,RMDC0     ;IS CYLINDER > 822??
12215  057252  103425                          BLO     110$            ;YES!!
12216  057254  042737  040000  061672          BIC     #SKI,500$       ;RESET SKI FLAG
12217  057262  122737  000004  001405          CMPB    #4,RMDA0+1      ;IS TRACK > 4??
12218  057270  103416                          BLO     110$            ;YES!!
12219  057272  122737  000037  001404          CMPB    #31.,RMDA0      ;IS SECTOR > 31??
12220  057300  103412                          BLO     110$            ;YES!!
12221  057302  122737  000035  001404          CMPB    #29.,RMDA0      ;IS SECTOR > 29??
12222  057310  103004                          BHIS    105$            ;NO - IAE SHOULD BE ZERO
12223  057312  032737  010000  001430          BIT     #FMT16,RMOF0    ;18 BIT FORMAT??
12224  057320  001402                          BEQ     110$            ;YES!!
12225  057322  005037  001140          105$:   CLR     $GDDAT          ;IAE SHOULD BE ZERO
12226  057326  013737  001342  001142  110$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
12227  057334  042737  175777  001142          BIC     #^CIAE,$BDDAT
12228  057342  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "IAE" STATUS OK??
12229  057350  001004                          BNE     115$            ;NO!!
12230  057352  042737  040000  061672          BIC     #SKI,500$       ;IAE OK - SKI SHOULD BE 0
12231  057360  000412                          BR      120$
12232  057362  062716  000004          115$:   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12233  057366  112776  000305  000000          MOVB    #305,@(SP)      ;WRITE ERROR NUMBER
12234  057374  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12235  057400  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12236  057402  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12237  057406                          120$:
12238
12239                                  ;REPORT AN ERROR IF "SKI" IS SET AND "IAE" STATUS WAS OK
12240  057406  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12241  057414  042737  137777  001142          BIC     #^CSKI,$BDDAT
12242  057422  013737  061672  001140          MOV     500$,$GDDAT     ;EXPECTED STATUS
12243  057430  042737  137777  001140          BIC     #^CSKI,$GDDAT
12244  057436  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI" SET??
12245  057444  001417                          BEQ     140$            ;NO!!
12246  057446  032737  040000  061672          BIT     #SKI,500$       ;WAS SKI CAUSED BY IAE = 0??
12247  057454  001032                          BNE     150$            ;YES - DON'T REPORT SKI
12248  057456  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12249  057462  112776  000306  000000          MOVB    #306,@(SP)      ;WRITE ERROR NUMBER
12250  057470  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12251  057474  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12252  057476  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12253  057502  000417                          BR      150$
12254
12255  057504                          140$:
12256
12257                                  ;REPORT AN ERROR IF SKI = 0 AND IAE WAS NOT DETECTED
12258  057504  032737  040000  061672          BIT     #SKI,500$       ;SHOULD SKI BE SET??
12259  057512  001413                          BEQ     150$            ;NO!!
12260  057514  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12261  057520  112776  000307  000000          MOVB    #307,@(SP)      ;WRITE ERROR NUMBER IN CALL
12262  057526  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12263  057532  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12264  057534  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
```

```
12265  057540  000240                          NOP
12266  057542                          150$:
12267
12268                                  ;LOOK FOR "LSC" OR "LBC" OR "DVC" IN ERROR REGISTER #2
12269  057542  032737  006200  001370          BIT     #LSC!LBC!DVC,RMER2I
12270  057550  001512                          BEQ     180$            ;NO ERRORS SET
12271
12272                                  ;REPORT ANY DEVICE FAULT, I.E., "DVC" = 1
12273  057552  032737  000200  001370          BIT     #DVC,RMER2I     ;IS "DVC" = 1??
12274  057560  001424                          BEQ     160$            ;NO!!
12275  057562  013737  001370  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12276  057570  042737  000200  001140          BIC     #DVC,$GDDAT
12277  057576  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12278  057604  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
12279  057610  112776  000310  000000          MOVB    #310,@(SP)      ;WRITE ERROR NUMBER IN CALL
12280  057616  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12281  057622  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12282  057624  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12283  057630  000240                          NOP
12284  057632                          160$:
12285
12286                                  ;REPORT LOSS OF BIT CLOCK, I.E.; "LBC" = 1, IF "MOL" = 1
12287  057632  032737  002000  001370          BIT     #LBC,RMER2I     ;IS LBC SET??
12288  057640  001430                          BEQ     170$            ;NO!!
12289  057642  032737  010000  001340          BIT     #MOL,RMDSI      ;WAS LBC ERROR BY MOL = 0
12290  057650  001424                          BEQ     170$            ;YES!!
12291  057652  013737  001370  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12292  057660  042737  002000  001140          BIC     #LBC,$GDDAT
12293  057666  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12294  057674  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12295  057700  112776  000311  000000          MOVB    #311,@(SP)      ;WRITE ERROR NUMBER IN CALL
12296  057706  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12297  057712  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12298  057714  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12299  057720  000240                          NOP
12300  057722                          170$:
12301
12302                                  ;REPORT LOS OF SYSTEM CLOCK, I.E., "LSC" = 1
12303  057722  032737  004000  001370          BIT     #LSC,RMER2I     ;IS "LSC" = 1??
12304  057730  001422                          BEQ     180$            ;NO!!
12305  057732  013737  001370  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12306  057740  042737  004000  001140          BIC     #LSC,$GDDAT
12307  057746  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12308  057754  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12309  057760  112776  000312  000000          MOVB    #312,@(SP)      ;WRITE ERROR NUMBER
12310  057766  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12311  057770  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12312  057774  000240                          NOP
12313  057776                          180$:
12314
12315                                  ;LOOK FOR "UNS" OR "DTE" OR "WLE" IN ERROR REGISTER #1
12316  057776  032737  054000  001342          BIT     #UNS!DTE!WLE,RMER1I
12317  060004  001527                          BEQ     220$            ;NO BITS SET
12318                                  ;REPORT "UNS" ERROR IF "DVC" = 0
12319  060006  032737  040000  001342          BIT     #UNS,RMER1I     ;IS "UNS" SET??
12320  060014  001427                          BEQ     190$            ;NO!!
```

```
12321  060016  032737  000200  001370          BIT     #DVC,RMER2I     ;WAS "UNS" CAUSED BY "DVC"??
12322  060024  001023                          BNE     190$            ;YES!!
12323  060026  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12324  060034  042737  040000  001140          BIC     #UNS,$GDDAT
12325  060042  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12326  060050  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12327  060054  112776  000313  000000          MOVB    #313,@(SP)      ;WRITE ERROR NUMBER
12328  060062  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12329  060066  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12330  060070  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12331  060074                          190$:
12332
12333                                  ;REPORT ANY DRIVE TIMING ERROR, I.E., "DTE" = 1
12334  060074  032737  010000  001342          BIT     #DTE,RMER1I     ;IS DTE SET??
12335  060102  001423                          BEQ     200$            ;NO!!
12336  060104  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12337  060112  042737  010000  001140          BIC     #DTE,$GDDAT
12338  060120  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12339  060126  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12340  060132  112776  000314  000000          MOVB    #314,@(SP)      ;WRITE ERROR NUMBER IN CALL
12341  060140  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12342  060144  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12343  060146  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12344  060152                          200$:
12345
12346                                  ;REPORT AN ERROR IF WRITE LOCK ERROR IS SET.  SEE IF DRIVE IS NOT
12347                                  ;WRITE PROTECTED, OR IF FUNCTION WAS NOT A WRITE
12348  060152  032737  004000  001342          BIT     #WLE,RMER1I     ;WAS "WLE" SET??
12349  060160  001441                          BEQ     220$            ;NO!!
12350  060162  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12351  060170  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12352  060176  052737  004000  001140          BIS     #WLE,$GDDAT
12353  060204  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12354  060210  112776  000315  000000          MOVB    #315,@(SP)      ;WRITE ERROR NUMBER IN CALL
12355  060216  032737  004000  001340          BIT     #WRL,RMDSI      ;WAS DRIVE WRITE PROTECTED??
12356  060224  001404                          BEQ     205$            ;NO!!
12357  060226  032737  000010  001376          BIT     #BIT3,RMCS10    ;WAS COMMAND A WRITE??
12358  060234  001406                          BEQ     210$            ;YES!!
12359  060236  112776  000316  000000  205$:   MOVB    #316,@(SP)      ;CHANGE ERROR NUMBER
12360  060244  042737  004000  001140          BIC     #WLE,$GDDAT
12361  060252  162716  000002          210$:   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12362  060256  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12363  060260  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12364
12365  060264                          220$:
12366
12367                                  ;OMIT DATA ERROR CHECKS IF ANY PREVIOUS ERRORS HAVE BEEN DETECTED
12368  060264  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12369  060270  105776  000000                  TSTB    @(SP)           ;WAS ERROR DETECTED??
12370  060274  001404                          BEQ     225$            ;NO - DO DATA CHECKS
12371  060276  162716  000004                  SUB     #4,(SP)         ;RESTORE SP
12372  060302  000137  061304                  JMP     340$            ;SKIP DATA CHECKS
12373  060306  162716  000004          225$:   SUB     #4,(SP)         ;RESTORE SP
12374
12375                                  ;CHECK HEADER ERRORS IF FUNCTION WAS NOT WRITE HEADER AND DATA, AND
12376                                  ;IF HEADER COMPARE IS NOT INHIBITED
```

```
060312  013737  001376  061674          MOV     RMCS10,510$     ;STRIP AND STORE FUNCTION CODE
060320  042737  177700  061674          BIC     #↑CFNCMSK,510$
060326  022737  000063  061674          CMP     #WH!GO,510$     ;WAS FUNCTION WRITE HEADER & DATA??
060334  001512                          BEQ     250$            ;YES - SKIP HEADER CHECKS
060336  032737  002000  001360          BIT     #HCI,RMOFI      ;WAS HCI SET??
060344  001106                          BNE     250$            ;YES - SKIP HEADER CHECKS

                                ;SEE IF ANY HEADER ERRORS ARE SET, I.E., "FER" OR "HCRC" OR "HCE"
060346  032737  000620  001342          BIT     #HCRC!FER!HCE,RMER1I
060354  001533                          BEQ     270$            ;NO ERRORS SET

                                ;REPORT HEADER CRC ERROR IF SET
060356  032737  000400  001342          BIT     #HCRC,RMER1I    ;WAS HCRC SET??
060364  001422                          BEQ     230$            ;NO!!
060366  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
060374  042737  000400  001140          BIC     #HCRC,$GDDAT
060402  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
060410  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
060414  112776  000317  000000          MOVB    #317,@(SP)      ;WRITE ERROR NUMBER
060422  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
060426  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
060430  000501                          BR      260$
060432                          230$:

                                ;REPORT FORMAT ERROR IF SET
060432  032737  000020  001342          BIT     #FER,RMER1I     ;WAS "FER" SET??
060440  001422                          BEQ     240$            ;NO!!
060442  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
060450  042737  000020  001140          BIC     #FER,$GDDAT
060456  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
060464  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
060470  112776  000320  000000          MOVB    #320,@(SP)      ;WRITE ERROR NUMBER
060476  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
060502  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
060504  000453                          BR      260$
060506                          240$:

                                ;REPORT HEADER COMPARE ERROR IF SET
060506  032737  000200  001342          BIT     #HCE,RMER1I     ;WAS "HCE" SET??
060514  001453                          BEQ     270$            ;NO!!
060516  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
060524  042737  000200  001140          BIC     #HCE,$GDDAT
060532  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
060540  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
060544  112776  000321  000000          MOVB    #321,@(SP)      ;WRITE ERROR NUMBER
060552  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
060556  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
060560  000425                          BR      260$
                                ;THERE SHOULD BE NO HEADER ERRORS BECAUSE
                                ;       .COMMAND WAS WRITE HEADER AND DATA, OR
                                ;       .HEADER COMPARE INHIBIT WAS SET
060562  032737  000620  001342  250$:   BIT     #HCE!FER!HCRC,RMER1I
060570  001425                          BEQ     270$            ;NO ERRORS WERE SET
060572  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
060600  042737  000620  001140          BIC     #HCE!FER!HCRC,$GDDAT
060606  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
```

```
12433  060614  062716  000004            ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
12434  060620  112776  000322  000000    MOVB    #322,@(SP)       ;WRITE ERROR NUMBER
12435  060626  162716  000002            SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
12436  060632  004736                     JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
12437  060634  162716  000010    260$:   SUB     #10,(SP)         ;MOVE SP TO NO ERROR
12438  060640  000137  061304            JMP     340$             ;OMIT FURTHER DATA CHECKS
12439
12440  060644                    270$:
12441
12442                             ;IF COMMAND WAS A WRITE COMMAND, GO DO WRITE ERROR CHECKS, OTHERWISE
12443                             ;DO READ ERROR CHECKS
12444  060644  032737  000010  061674      BIT     #BIT3,510$       ;WAS THIS A WRITE COMMAND?
12445  060652  001002                      BNE     275$             ;NO!!
12446  060654  000137  061072              JMP     310$             ;GO DO WRITE STATUS CHECK
12447  060660                    275$:
12448
12449                             ;REPORT DATA CHECK IF SET
12450  060660  032737  100000  001342      BIT     #DCK,RMER1I      ;DATA CHECK ERROR??
12451  060666  001450                      BEQ     290$             ;NO!!
12452  060670  013737  001342  001140      MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
12453  060676  042737  100000  001140      BIC     #DCK,$GDDAT
12454  060704  013737  001342  001142      MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
12455  060712  062716  000004              ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR
12456  060716  112776  000323  000000      MOVB    #323,@(SP)       ;WRITE ERROR NUMBER
12457  060724  032737  004000  001360      BIT     #ECI,RMOFI       ;WAS ECC CORRECTION DISABLED??
12458  060732  001021                      BNE     280$             ;YES!!
12459  060734  112776  000324  000000      MOVB    #324,@(SP)       ;CHANGE TO RECOVERABLE ERROR
12460  060742  032737  000100  001342      BIT     #ECH,RMER1I      ;IS ERROR RECOVERABLE??
12461  060750  001007                      BNE     276$             ;NO !!
12462                             ;DO NOT REPORT RECOVERABLE ERROR IF READ COMMAND
12463  060752  032737  000020  061674      BIT     #BIT4,510$       ;WAS THIS A READ COMMAND ??
12464  060760  001406                      BEQ     280$             ;NO !!
12465  060762  162716  000004              SUB     #4,(SP)          ;RESTORE SP
12466  060766  000410                      BR      290$             ;SKIP ERROR - DATA WILL BE CORRECTED
12467  060770  112776  000325  000000  276$:   MOVB    #325,@(SP)       ;CHANGE TO NON RECOVERABLE
12468  060776  162716  000002            280$:   SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
12469  061002  004736                     JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
12470  061004  162716  000010             SUB     #10,(SP)         ;RESTORE SP TO NO ERROR
12471
12472  061010                    290$:
12473
12474                             ;REPORT DATA BUS PARITY ERROR IF SET, I.E., MDPE = 1
12475  061010  032737  000400  001336      BIT     #MDPE,RMCS2I     ;PARITY ERROR SET??
12476  061016  001423                      BEQ     300$             ;NO!!
12477  061020  013737  001336  001140      MOV     RMCS2I,$GDDAT    ;EXPECTED STATUS
12478  061026  042737  000400  001140      BIC     #MDPE,$GDDAT
12479  061034  013737  001336  001142      MOV     RMCS2I,$BDDAT    ;RECEIVED STATUS
12480  061042  062716  000004              ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR
12481  061046  112776  000326  000000      MOVB    #326,@(SP)       ;WRITE ERROR NUMBER
12482  061054  162716  000002              SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
12483  061060  004736                      JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
12484  061062  162716  000010              SUB     #10,(SP)         ;MOVE SP TO NO ERROR
12485  061066  000137  061304    300$:     JMP     340$             ;SKIP WRITE STATUS CHECK
12486
12487  061072                    310$:
12488
```

```
12489
12490  061072  032737  000001  001340          ;TEST TO SEE THAT OFFSET MODE WAS RESET; REPORT ERROR IF "OM" = 1
12491  061100  001423                  BIT     #OM,RMDSI       ;IS OFFSET ON??
12492  061102  013737  001340  001140  BEQ     320$            ;NO
12493  061110  042737  000001  001140  MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12494  061116  013737  001340  001142  BIC     #OM,$GDDAT
12495  061124  062716  000004          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12496  061130  112776  000327  000000  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12497  061136  162716  000002          MOVB    #327,@(SP)      ;WRITE ERROR NUMBER IN CALL
12498  061142  004736                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12499  061144  162716  000010          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12500  061150                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12501                          320$:
12502                          ;TEST FOR DATA BUS PARITY ERROR; REPORT ERROR IF "DPE" = 1
12503  061150  032737  000010  001370          BIT     #DPE,RMER2I     ;DATA PARITY ERROR??
12504  061156  001423                  BEQ     330$            ;NO!!
12505  061160  013737  001370  001140  MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12506  061166  042737  000010  001140  BIC     #DPE,$GDDAT
12507  061174  013737  001370  001142  MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12508  061202  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12509  061206  112776  000330  000000  MOVB    #330,@(SP)      ;WRITE ERROR NUMBER
12510  061214  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12511  061220  004736                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12512  061222  162716  000010          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12513  061226                          330$:
12514
12515                          ;TEST FOR WRITE CLOCK FAILURE; REPORT ERROR IF "WCF" = 1
12516  061226  032737  000040  001342          BIT     #WCF,RMER1I     ;IS "WCF" SET??
12517  061234  001423                  BEQ     340$            ;NO!!
12518  061236  013737  001342  001140  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12519  061244  042737  000040  001140  BIC     #WCF,$GDDAT
12520  061252  013737  001342  001142  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12521  061260  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12522  061264  112776  000331  000000  MOVB    #331,@(SP)      ;WRITE ERROR NUMBER
12523  061272  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12524  061276  004736                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12525  061300  162716  000010          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12526  061304                          340$:
12527
12528                          ;REPORT "DATA LATE" ERROR IF "DLT" = 1
12529  061304  032737  100000  001336          BIT     #DLT,RMCS2I     ;IS "DLT" SET??
12530  061312  001423                  BEQ     350$            ;NO!!
12531  061314  013737  001336  001140  MOV     RMCS2I,$GDDAT   ;EXPECTED STATUS
12532  061322  042737  100000  001140  BIC     #DLT,$GDDAT
12533  061330  013737  001336  001142  MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
12534  061336  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12535  061342  112776  000332  000000  MOVB    #332,@(SP)      ;WRITE ERROR NUMBER
12536  061350  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12537  061354  004736                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12538  061356  162716  000010          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12539  061362                          350$:
12540                          ;LOOK FOR UNEXPECTED CHANGES IN DRIVE STATUS
12541  061362  013746  001340          MOV     RMDSI,-(SP)     ;STACK DRIVE STATUS
12542  061366  042716  147677          BIC     #^C<PIP!MOL!VV>,(SP) ;CLEAR DONT CARES
12543  061372  022726  010100          CMP     #MOL!VV,(SP)+   ;IS DRIVE STATUS OK??
12544  061376  001522                  BEQ     380$            ;YES!!
```

```
12545
12546                                          ;REPORT ERROR IF POSITIONING IN PROGRESS AND NO SEEK INCOMPLETE ERROR,
12547                                          ;I.E, PIP = 1 AND SKI = 0
12548  061400  032737  020000  001340                BIT     #PIP,RMDSI      ;IS "PIP" SET??
12549  061406  001430                              BEQ     360$            ;NO!!
12550  061410  032737  040000  001370              BIT     #SKI,RMER2I     ;WAS "SKI" ERROR REPORTED??
12551  061416  001024                              BNE     360$            ;YES-DONT REPORT PIP
12552  061420  013737  001340  001140              MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12553  061426  042737  020000  001140              BIC     #PIP,$GDDAT
12554  061434  013737  001340  001142              MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12555  061442  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12556  061446  112776  000333  000000              MOVB    #333,@(SP)      ;WRITE ERROR NUMBER
12557  061454  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12558  061460  004736                              JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12559  061462  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12560  061466  000240                              NOP
12561  061470                                 360$:
12562
12563                                          ;REPORT ERROR IF MEDIUM IS NOT ON LINE AND OPI ERROR WAS NOT
12564                                          ;REPORTED, I.E., MOL = OPI = 0
12565  061470  032737  010000  001340                BIT     #MOL,RMDSI      ;IS MEDIUM ON LINE??
12566  061476  001027                              BNE     370$            ;YES!!
12567  061500  032737  020000  001342              BIT     #OPI,RMER1I     ;WAS OPI ERROR REPORTED??
12568  061506  001023                              BNE     370$            ;YES!!
12569  061510  013737  001340  001140              MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12570  061516  052737  010000  001140              BIS     #MOL,$GDDAT
12571  061524  013737  001340  001142              MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12572  061532  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12573  061536  112776  000334  000000              MOVB    #334,@(SP)      ;WRITE ERROR NUMBER
12574  061544  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12575  061550  004736                              JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12576  061552  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12577  061556                                 370$:
12578
12579                                          ;REPORT ERROR IF VOLUME IS NOT VALID AND "IVC" ERROR WAS NOT
12580                                          ;REPORTED, I.E., VV = IVC = 0
12581  061556  032737  000100  001340                BIT     #VV,RMDSI       ;IS VOLUME VALID??
12582  061564  001027                              BNE     380$            ;YES!!
12583  061566  032737  010000  001370              BIT     #IVC,RMER2I     ;WAS IVC ERROR REPORTED??
12584  061574  001033                              BNE     390$            ;YES!!
12585  061576  013737  001340  001140              MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12586  061604  052737  000100  001140              BIS     #VV,$GDDAT
12587  061612  013737  001340  001142              MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12588  061620  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12589  061624  112776  000335  000000              MOVB    #335,@(SP)      ;WRITE ERROR NUMBER
12590  061632  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12591  061636  004736                              JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12592  061640  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12593  061644                                 380$:
12594
12595                                          ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS FOUND
12596  061644  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12597  061650  105776  000000                      TSTB    @(SP)           ;ANY ERROR??
12598  061654  001403                              BEQ     390$            ;NO!!
12599  061656  062716  000004                      ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
12600  061662  000402                              BR      400$
```

```
12601  061664  162716  000004          390$:   SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
12602
12603  061670  000207                  400$:   RTS     PC              ;RETURN TO USER
12604
12605  061672  000000                  500$:   .WORD                   ;ERROR FLAGS
12606  061674  000000                  510$:   .WORD                   ;TEMPORARY STORAGE
```

```
12607                                   .SBTTL  STATIC DRIVE STATUS CHECK SUBROUTINE
12608
12609                                   ;THIS SUBROUTINE LOOKS FOR UNEXPECTED CHANGES IN DRIVE
12610                                   ;STATUS, SUCH AS THE DRIVE LOSING VOLUME VALID.  THE SUBROUTINE
12611                                   ;CAN BE USED BY HOUSEKEEPING AND OTHER COMMANDS DURING WHICH THERE
12612                                   ;SHOULD NOT BE ANY DRIVE ERRORS OR CHANGES IN STATE.
12613
12614                                   ;THE FOLLOWING CONDITIONS ARE TESTED AND REPORTED AS ERRORS
12615                                   ;IF TRUE:
12616
12617                                   ;       .MOL = 0, INDICATES DRIVE WENT OFFLINE, NOTE
12618                                   ;THAT MOL IS ASSUMED TO HAVE BEEN SET
12619                                   ;       .VV = 0, INDICATES THE DRIVE LOST VOLUME VALID
12620                                   ;       .PIP = 1, INDICATES THAT THE DRIVE IS OFF CYLINDER
12621                                   ;       .SKI = 1, INDICATES THE DRIVE HAS AN UNEXPECTED SKI ERROR
12622                                   ;       .DVC = 1, INDICATES AN UNEXPECTED DEVICE FAULT
12623
12624                                   ;THE SUBROUTINE IS CALLED AFTER STORING STATUS IN THE GET BUFFER.
12625
12626                                   ;(1)    JSR     PC,STCDRVSTS
12627                                   ;       BR      ???                     RETURN HERE IF NO ERROR
12628                                   ;       NOP                             RETURN HERE TO REPORT AN ERROR
12629                                   ;       ERROR                           ERROR NUMBER DEFINED BY SUB
12630                                   ;       JSR     PC,@(SP)+               GO BACK TO SUB FOR MORE ERROR CHECKS
12631                                   ;       ???                             RETURN HERE IF NO MORE ERRORS
12632
12633  061676                           STCDRVSTS:
12634
12635                                   ;CLEAR USER'S ERROR CALL
12636  061676  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
12637  061702  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
12638  061706  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR RETURN
12639                                   ;SEE IF "MOL" = "VV" = 1, AND "PIP" = 0
12640  061712  013746  001340                  MOV     RMDSI,-(SP)     ;PUT DRIVE STATUS ON STACK
12641  061716  042716  147677                  BIC     #^C<PIP!MOL!VV>,(SP)
12642  061722  022726  010100                  CMP     #MOL!VV,(SP)+   ;ARE MOL,VV AND PIP O.K.??
12643  061726  001524                          BEQ     30$             ;YES!!
12644
12645                                   ;REPORT AN ERROR IF MOL = 0 AND "OPI" = 0
12646  061730  032737  010000  001340          BIT     #MOL,RMDSI      ;IS MOL ON ??
12647  061736  001030                          BNE     10$             ;YES!!
12648  061740  032737  020000  001342          BIT     #OPI,RMER1I     ;WAS "OPI" SET??
12649  061746  001024                          BNE     10$             ;YES-DONT REPORT "MOL" = 0
12650  061750  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12651  061756  052737  010000  001140          BIS     #MOL,$GDDAT
12652  061764  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12653  061772  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12654  061776  112776  000207  000000          MOVB    #207,@(SP)      ;WRITE ERROR NUMBER IN CALL
12655  062004  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12656  062010  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12657  062012  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
```

```
12658  062016  000240                          NOP
12659  062020                                  10$:
12660
12661                                          ;REPORT AN ERROR IF VOLUME VALID IS NOW ZERO AND "IVC" = 0
12662  062020  032737  000100  001340          BIT     #VV,RMDSI       ;IS "VV" = 0??
12663  062026  001030                          BNE     20$             ;NO!!
12664  062030  032737  010000  001370          BIT     #IVC,RMER2I     ;WAS "IVC" SET??
12665  062036  001024                          BNE     20$             ;YES-DONT REPORT "VV" = 0
12666  062040  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12667  062046  052737  000100  001340          BIS     #VV,RMDSI
12668  062054  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12669  062062  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
12670  062066  112776  000210  000000          MOVB    #210,@(SP)      ;WRITE ERROR NUMBER IN CALL
12671  062074  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12672  062100  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12673  062102  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
12674  062106  000240                          NOP
12675  062110                                  20$:
12676
12677                                          ;REPORT AN ERROR IF DRIVE IS OFF CYLINDER AND "SKI" = 0
12678  062110  032737  020000  001340          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
12679  062116  001430                          BEQ     30$             ;NO!!
12680  062120  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI" SET??
12681  062126  001024                          BNE     30$             ;YES-DONT REPORT "PIP" = 1
12682  062130  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12683  062136  042737  020000  001140          BIC     #PIP,$GDDAT
12684  062144  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12685  062152  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12686  062156  112776  000211  000000          MOVB    #211,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
12687  062164  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12688  062170  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12689  062172  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12690  062176  000240                          NOP
12691  062200                                  30$:
12692
12693                                          ;SEE IF "SKI" = "DVC" = 0
12694  062200  013746  001370                  MOV     RMER2I,-(SP)    ;PUT ERROR REG 2 ON STACK
12695  062204  042726  137577                  BIC     #^C<SKI!DVC>,(SP)+
12696  062210  001460                          BEQ     60$             ;BRANCH IF NO ERROR
12697  062212                                  40$:
12698
12699                                          ;REPORT AN ERROR IF THERE IS A DEVICE FAULT
12700  062212  032737  000200  001370          BIT     #DVC,RMER2I     ;ANY DEVICE FAULT??
12701  062220  001424                          BEQ     50$             ;NO!!
12702  062222  013737  001370  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12703  062230  042737  000200  001140          BIC     #DVC,$GDDAT
12704  062236  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12705  062244  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S CALL
12706  062250  112776  000212  000000          MOVB    #212,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12707  062256  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12708  062262  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12709  062264  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
12710  062270  000240                          NOP
12711  062272                                  50$:
12712
12713                                          ;REPORT AN ERROR IF "SKI" = 1
```

```
12714  062272  032737  040000  001370          BIT     #SKI,RMER2I     ;IS THERE A SEEK INCOMPLETE ERROR
12715  062300  001424                          BEQ     60$             ;NO!!
12716  062302  013737  001370  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12717  062310  042737  040000  001140          BIC     #SKI,$GDDAT
12718  062316  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12719  062324  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12720  062330  112776  000213  000000          MOVB    #213,@(SP)      ;WRITE ERROR NUMBER IN USER'S ERROR CALL
12721  062336  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12722  062342  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12723  062344  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
12724  062350  000240                          NOP
12725  062352                          60$:
12726
12727                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
12728  062352  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12729  062356  105776  000000                  TSTB    @(SP)           ;WAS AN ERROR DETECTED??
12730  062362  001403                          BEQ     70$             ;NO!!
12731  062364  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO USER'S ERROR RETURN
12732  062370  000402                          BR      80$
12733  062372  162716  000004          70$:    SUB     #4,(SP)         ;NO - MOVE SP TO NO ERROR RETURN
12734  062376  000240                  80$:    NOP
12735  062400  000207                          RTS     PC              ;RETURN TO USER
```

```
12736                                   .SBTTL  COMPOSITE ERROR CHECK SUBROUTINE
12737
12738                                   ;THIS SUBROUTINE CHECKS THE STORED CONTENTS OF RMER1 AND
12739                                   ;RMER2 AFTER MASKING EACH REGISTER WORD WITH THE USER'S STATUS
12740                                   ;MASKS AND REPORTS AN ERROR IF ANY BITS ARE LEFT ON AFTER
12741                                   ;THE MASKS ARE APPLIED.
12742
12743                                   ;CALL:
12744                                   ;(1)    JSR     PC,CMPERRSTS
12745                                   ;       .WORD                   MASK FOR ERROR REGISTER 1
12746                                   ;       .WORD                   MASK FOR ERROR REGISTER 2
12747                                   ;       BR      ???             RETURN HERE IF NO ERROR
12748                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
12749                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
12750                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
12751                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
12752
12753                                   ;NOTE:  BITS TO BE MASKED SHOULD BE ONE; BITS TO BE TESTED SHOULD
12754                                   ;BE ZERO
12755
12756  062402                           CMPERRSTS:
12757
12758                                   ;MASK AND STORE THE CONTENTS OF RMER1  AND RMER2
12759  062402  013737  001342  001176           MOV     RMER1I,$TMP1    ;STORE RMER1 AT TEMP STORAGE
12760  062410  047637  000000  001176           BIC     @(SP),$TMP1     ;MASK   RMER1
12761  062416  062716  000002                   ADD     #2,(SP)         ;MOVE SP TO NEXT MASK
12762  062422  013737  001370  001200           MOV     RMER2I,$TMP2    ;STORE RMER2 AT TEMP STORAGE
12763  062430  047637  000000  001200           BIC     @(SP),$TMP2     ;MASK RMER2
12764
12765
12766                                   ;CLEAR USER'S ERROR CALL
12767  062436  062716  000006                   ADD     #6,(SP)         ;MOVE SP TO USER'S ERROR CALL
12768  062442  105076  000000                   CLRB    @(SP)           ;CLEAR ERROR NUMBER
12769  062446  162716  000004                   SUB     #4,(SP)         ;LEAVE SP AT NO ERROR RETURN
12770
12771                                   ;SEE IF THERE WERE ANY ERRORS IN RMER1, I.E., $TMP1
12772  062452  005737  001176                   TST     $TMP1           ;ANY ERRORS TO REPORT??
12773  062456  001420                           BEQ     5$              ;NO !!
12774  062460  013737  001176  001142           MOV     $TMP1,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
12775  062466  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
12776  062472  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12777  062476  112776  000066  000000           MOVB    #66,@(SP)       ;CORRECTABLE DATA CHECK ERROR #
12778  062504  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12779  062510  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12780  062512  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
12781  062516  000240                           NOP
12782  062520                           5$:
12783
12784
12785                                   ;SEE IF THERE ARE ANY ERRORS TO REPORT IN RMER2 ($TMP2)
12786  062520  005737  001200                   TST     $TMP2           ;ANY ERRORS IN RMER2?
12787  062524  001420                           BEQ     10$             ;NO!!
12788
12789  062526  013737  001200  001142           MOV     $TMP2,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
12790  062534  005037  001140                   CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
12791  062540  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
```

```
12792  062544  112776  000067  000000          MOVB    #67,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
12793  062552  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12794  062556  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12795  062560  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12796  062564  000240                          NOP
12797  062566                          10$:
12798
12799                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
12800  062566  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12801  062572  105776  000000                  TSTB    @(SP)           ;WAS THERE AN ERROR CALLED??
12802  062576  001403                          BEQ     20$             ;NO!!
12803  062600  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
12804  062604  000402                          BR      30$
12805  062606  162716  000004          20$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
12806  062612  000207                  30$:    RTS     PC              ;RETURN TO USER
12807
12808
```

```
12809                                   .SBTTL  STOP AND SHUTDOWN SUBROUTINES
12810
12811  062614                           STOP:
12812
12813                                   ;DROP PRIORITY TO ALLOW CONSOLE INTERRUPT
12814  062614  012746  000140                   MOV     #PR3,-(SP)      ;;PUT NEW PS ON STACK
12815  062620  012746  062626                   MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
12816  062624  000002                           RTI                     ;;POP NEW PC AND PS
12817  062626                           64$:
12818  062626  000240                           NOP
12819                                   ;RAISE PRIORITY TO INHIBIT INTERRUPT
12820  062630  012746  000300                   MOV     #PR6,-(SP)      ;;PUT NEW PS ON STACK
12821  062634  012746  062642                   MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
12822  062640  000002                           RTI                     ;;POP NEW PC AND PS
12823  062642                           65$:
12824
12825  062642  000207                   10$:    RTS     PC              ;CONTINUE
12826
12827
12828  062644                           SHUT:
12829  062644  104401  062666                   TYPE    ,100$           ;TYPE THE HALT MESSAGE
12830  062650  005737  000042                   TST     42              ;RUNNING STANDALONE ??
12831  062654  001402                           BEQ     10$             ;YES !!
12832  062656  000137  036246                   JMP     $EOP            ;NO - GO TO END OF PASS
12833  062662                           10$:
12834  062662  000137  005322                   JMP     START           ;GO TO START
12835  062666  042524  052123  053440   100$:   .ASCIZ  @TEST WAS HALTED@<CR><LF><LF>
12836  062674  051501  044040  046101
12837  062702  042524  006504  005012
12838  062710     000
12839          062712                   .EVEN
```

```
                                .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

                                ;;******************************************************************
                                ;*SAVE R0-R5
                                ;*CALL:
                                ;*      SAVREG
                                ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
                                ;*
                                ;*TOP---(+16)
                                ;* +2---(+18)
                                ;* +4---R5
                                ;* +6---R4
                                ;* +8---R3
                                ;*+10---R2
                                ;*+12---R1
                                ;*+14---R0
                                ;
062712                          $SAVREG:
010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
010446                                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
010546                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
000002                                  RTI
                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
062750                          $RESREG:
012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
012605                                  MOV     (SP)+,R5        ;;POP STACK INTO R5
012604                                  MOV     (SP)+,R4        ;;POP STACK INTO R4
012603                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
000002                                  RTI
                                .SBTTL  BINARY TO ASCII AND TYPE ROUTINE

                                ;;******************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
                                ;*BINARY-ASCII NUMBER AND TYPE IT.
                                ;*CALL:
                                ;*      MOV     NUMBER,-(SP)    ;;NUMBER TO BE TYPED
                                ;*      TYPBN                   ;;TYPE IT

010146                          $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
016601  000006                          MOV     6(SP),R1        ;;GET THE INPUT NUMBER
```

```
12896  063014  000261                          SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
12897  063016  112737  000060  063060  1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
12898  063024  006101                          ROL     R1              ;;GET THIS BIT
12899  063026  001406                          BEQ     2$              ;;DONE?
12900  063030  105537  063060                  ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
12901  063034  104401  063060                  TYPE    ,$BIN           ;;GO TYPE THIS BIT
12902  063040  000241                          CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
12903  063042  000765                          BR      1$              ;;GO DO THE NEXT BIT
12904  063044  012601                  2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
12905  063046  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
12906  063054  012616                          MOV     (SP)+,(SP)
12907  063056  000002                          RTI                     ;;RETURN TO USER
12908  063060      000      000         $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
12909                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
12910
12911                                  ;;*****************************************************************
12912                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
12913                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
12914                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
12915                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
12916                                  ;*REPLACED WITH SPACES.
12917                                  ;*CALL:
12918                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
12919                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
12920
12921  063062                          $TYPDS:
12922  063062  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12923  063064  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12924  063066  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12925  063070  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
12926  063072  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
12927  063074  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
12928  063100  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
12929  063104  100004                          BPL     1$              ;;BR IF INPUT IS POS.
12930  063106  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
12931  063110  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
12932  063116  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
12933  063120  012703  063276                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
12934  063124  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
12935  063130  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
12936  063132  016001  063266                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
12937  063136  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
12938  063140  002402                          BLT     4$              ;;BR IF DONE
12939  063142  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
12940  063144  000774                          BR      3$
12941  063146  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
12942  063150  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
12943  063152  001002                          BNE     5$              ;;FALL THROUGH IF 0
12944  063154  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
12945  063156  100407                          BMI     7$              ;;BR IF YES
12946  063160  106316                  5$:     ASLB    (SP)            ;;MSD?
12947  063162  103003                          BCC     6$              ;;BR IF NO
12948  063164  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
12949  063172  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
12950  063176  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
12951  063202  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
```

```
12952  063204  005720                    TST     (R0)+           ;;JUST INCREMENTING
12953  063206  020027  000010            CMP     R0,#10          ;;CHECK THE TABLE INDEX
12954  063212  002746                    BLT     2$              ;;GO DO THE NEXT DIGIT
12955  063214  003003                    BGT     8$              ;;GO TO EXIT
12956  063216  010502                    MOV     R5,R2           ;;GET THE LSD
12957  063220  000764                    BR      6$              ;;GO CHANGE TO ASCII
12958  063222  105726            8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
12959  063224  100003                    BPL     9$              ;;BR IF NO
12960  063226  116663  177777  177776    MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
12961  063234  105013            9$:     CLRB    (R3)            ;;SET THE TERMINATOR
12962  063236  012605                    MOV     (SP)+,R5        ;;POP STACK INTO R5
12963  063240  012603                    MOV     (SP)+,R3        ;;POP STACK INTO R3
12964  063242  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
12965  063244  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
12966  063246  012600                    MOV     (SP)+,R0        ;;POP STACK INTO R0
12967  063250  104401  063276            TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
12968  063254  016666  000002  000004    MOV     2(SP),4(SP)     ;;ADJUST THE STACK
12969  063262  012616                    MOV     (SP)+,(SP)
12970  063264  000002                    RTI                     ;;RETURN TO USER
12971  063266  023420            $DTBL:  10000.
12972  063270  001750                    1000.
12973  063272  000144                    100.
12974  063274  000012                    10.
12975  063276  000004            $DBLK:  .BLKW   4
12976                             .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
12977
12978                             ;;**************************************************************
12979                             ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
12980                             ;*OCTAL (ASCII) NUMBER AND TYPE IT.
12981                             ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
12982                             ;*CALL:
12983                             ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
12984                             ;*      TYPOS                   ;;CALL FOR TYPEOUT
12985                             ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
12986                             ;*      .BYTE   M               ;;M=1 OR 0
12987                             ;*                                      ;;1=TYPE LEADING ZEROS
12988                             ;*                                      ;;0=SUPPRESS LEADING ZEROS
12989                             ;*
12990                             ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
12991                             ;*$TYPOS OR $TYPOC
12992                             ;*CALL:
12993                             ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
12994                             ;*      TYPON                   ;;CALL FOR TYPEOUT
12995                             ;*
12996                             ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
12997                             ;*CALL:
12998                             ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
12999                             ;*      TYPOC                   ;;CALL FOR TYPEOUT
13000
13001  063306  017646  000000    $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
13002  063312  116637  000001  063531  MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
13003  063320  112637  063533          MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
13004  063324  062716  000002          ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
13005  063330  000406                  BR      $TYPON
13006  063332  112737  000001  063531  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
13007  063340  112737  000006  063533          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
```

```
13008  063346  112737  000005  063530  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
13009  063354  010346                  MOV     R3,-(SP)        ;;SAVE R3
13010  063356  010446                  MOV     R4,-(SP)        ;;SAVE R4
13011  063360  010546                  MOV     R5,-(SP)        ;;SAVE R5
13012  063362  113704  063533          MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
13013  063366  005404                  NEG     R4
13014  063370  062704  000006          ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
13015  063374  110437  063532          MOVB    R4,$OMODE       ;;SAVE IT FOR USE
13016  063400  113704  063531          MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
13017  063404  016605  000012          MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
13018  063410  005003                  CLR     R3              ;;CLEAR THE OUTPUT WORD
13019  063412  006105          1$:     ROL     R5              ;;ROTATE MSB INTO "C"
13020  063414  000404                  BR      3$              ;;GO DO MSB
13021  063416  006105          2$:     ROL     R5              ;;FORM THIS DIGIT
13022  063420  006105                  ROL     R5
13023  063422  006105                  ROL     R5
13024  063424  010503                  MOV     R5,R3
13025  063426  006103          3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
13026  063430  105337  063532          DECB    $OMODE          ;;TYPE THIS DIGIT?
13027  063434  100016                  BPL     7$              ;;BR IF NO
13028  063436  042703  177770          BIC     #177770,R3      ;;GET RID OF JUNK
13029  063442  001002                  BNE     4$              ;;TEST FOR 0
13030  063444  005704                  TST     R4              ;;SUPPRESS THIS 0?
13031  063446  001403                  BEQ     5$              ;;BR IF YES
13032  063450  005204          4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
13033  063452  052703  000060          BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
13034  063456  052703  000040          BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
13035  063462  110337  063526  5$:     MOVB    R3,8$           ;;SAVE FOR TYPING
13036  063466  104401  063526          TYPE    ,8$             ;;GO TYPE THIS DIGIT
13037  063472  105337  063530  7$:     DECB    $OCNT           ;;COUNT BY 1
13038  063476  003347                  BGT     2$              ;;BR IF MORE TO DO
13039  063500  002402                  BLT     6$              ;;BR IF DONE
13040  063502  005204                  INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
13041  063504  000744                  BR      2$              ;;GO DO THE LAST DIGIT
13042  063506  012605          6$:     MOV     (SP)+,R5        ;;RESTORE R5
13043  063510  012604                  MOV     (SP)+,R4        ;;RESTORE R4
13044  063512  012603                  MOV     (SP)+,R3        ;;RESTORE R3
13045  063514  016666  000002  000004  MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
13046  063522  012616                  MOV     (SP)+,(SP)
13047  063524  000002                  RTI                     ;;RETURN
13048  063526     000  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
13049  063527     000          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
13050  063530     000  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
13051  063531     000  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
13052  063532  000000  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
13053                  .SBTTL  TYPE ROUTINE
13054
13055                  ;;*****************************************************************
13056                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
13057                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
13058                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
13059                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
13060                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
13061                  ;*
13062                  ;*CALL:
13063                  ;*1) USING A TRAP INSTRUCTION
```

```
13064                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
13065                                   ;*OR
13066                                   ;*      TYPE
13067                                   ;*      MESADR
13068                                   ;*
13069
13070  063534  105737  001173          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
13071  063540  100002                          BPL     1$              ;;BR IF YES
13072  063542  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
13073  063544  000430                          BR      3$              ;;LEAVE
13074  063546  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
13075  063550  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
13076  063554  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
13077  063562  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
13078  063564  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
13079  063572  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
13080  063574  010037  063604                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
13081  063600  004737  066444                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
13082  063604  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
13083  063606  132737  000040  001243  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
13084  063614  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
13085  063616  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
13086  063620  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
13087  063622  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
13088  063624  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
13089  063626  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
13090  063632  000002                          RTI                     ;;RETURN
13091  063634  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
13092  063640  001430                          BEQ     8$
13093  063642  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
13094  063646  001006                          BNE     5$
13095  063650  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
13096  063652  104401                          TYPE                    ;;TYPE A CR AND LF
13097  063654  001217                          $CRLF
13098  063656  105037  064012                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
13099  063662  000755                          BR      2$              ;;GET NEXT CHARACTER
13100  063664  004737  063746          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
13101  063670  123726  001172          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
13102  063674  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
13103  063676  013746  001170                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
13104                                                                  ;;AND THE NULL CHAR.
13105  063702  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
13106  063706  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
13107  063710  004737  063746                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
13108  063714  105337  064012                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
13109  063720  000770                          BR      7$              ;;LOOP
13110
13111                                   ;HORIZONTAL TAB PROCESSOR
13112
13113  063722  112716  000040          8$:     MOVB    #'  ,(SP)       ;;REPLACE TAB WITH SPACE
13114  063726  004737  063746          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
13115  063732  132737  000007  064012          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
13116  063740  001372                          BNE     9$              ;;TAB STOP
13117  063742  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
13118  063744  000724                          BR      2$              ;;GET NEXT CHARACTER
13119  063746  105777  115212          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
```

```
13120  063752  100375                        BPL     $TYPEC
13121  063754  116677  000002  115204        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
13122  063762  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
13123  063770  001003                        BNE     1$              ;;BRANCH IF NO
13124  063772  105037  064012                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
13125  063776  000406                        BR      $TYPEX          ;;EXIT
13126  064000  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
13127  064006  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
13128  064010  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
13129  064012  000000              $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
13130  064014  000207              $TYPEX: RTS     PC
13131
13132                               .SBTTL  SCOPE HANDLER ROUTINE
13133
13134                               ;;**************************************************************
13135                               ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
13136                               ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
13137                               ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
13138                               ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
13139                               ;*SW14=1        LOOP ON TEST
13140                               ;*SW11=1        INHIBIT ITERATIONS
13141                               ;*SW09=1        LOOP ON ERROR
13142                               ;*SW08=1        LOOP ON TEST IN SWR<7:0>
13143                               ;*CALL
13144                               ;*      SCOPE           ;;SCOPE=IOT
13145
13146  064016                       $SCOPE:
13147  064016  104410                        CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
13148  064020  004737  062614                JSR     PC,STOP
13149  064024  032777  040000  115122  1$:   BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
13150  064032  001131                        BNE     $OVER           ;;YES IF SW14=1
13151                               ;#####START OF CODE FOR THE XOR TESTER#####
13152  064034  000416              $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
13153                                                               ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
13154  064036  013746  000004                MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
13155  064042  012737  064062  000004        MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
13156  064050  005737  177060                TST     @#177060        ;;TIME OUT ON XOR?
13157  064054  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
13158  064060  000500                        BR      $SVLAD          ;;GO TO THE NEXT TEST
13159  064062  022626              5$:       CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
13160  064064  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
13161  064070  000440                        BR      7$              ;;LOOP ON THE PRESENT TEST
13162  064072                       6$:;#####END OF CODE FOR THE XOR TESTER#####
13163  064072  032777  000400  115054        BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
13164  064100  001421                        BEQ     2$              ;;BR IF NO
13165  064102  005046                        CLR     -(SP)           ;;CLEAR A TEMP. LOCATION
13166  064104  117716  115044                MOVB    @SWR,(SP)       ;;PICKUP THE DESIRED TEST NUMBER
13167  064110  001414                        BEQ     8$              ;;BRANCH IF BAD TEST NUMBER IN SWR
13168  064112  022716  000027                CMP     #27,(SP)        ;;CHECK THE NUMBER IN THE SWR
13169  064116  002411                        BLT     8$              ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
13170  064120  011637  001116                MOV     (SP),$TSTNM     ;;UPDATE THE TEST NUMBER
13171  064124  005316                        DEC     (SP)            ;;BACKUP BY ONE
13172  064126  006316                        ASL     (SP)            ;;SCALE THE TEST NUMBER AS AN INDEX
13173  064130  062716  064334                ADD     #$SW08TBL,(SP)  ;;FORM THE ADDRESS OF TEST POINTER
13174  064134  013637  001122                MOV     @(SP)+,$LPADR   ;;SET LOOP ADDRESS TO DESIRED TEST
13175  064140  000466                        BR      $OVER           ;;GO LOOP ON THE TEST
```

```
13176  064142  005726                  8$:     TST     (SP)+           ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
13177  064144  105737  001117          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
13178  064150  001421                          BEQ     3$              ;;BR IF NO
13179  064152  123737  001131  001117          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
13180  064160  101015                          BHI     3$              ;;BR IF NO
13181  064162  032777  001000  114764          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
13182  064170  001404                          BEQ     4$              ;;BR IF NO
13183  064172  013737  001124  001122  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
13184  064200  000446                          BR      $OVER
13185  064202  105037  001117          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
13186  064206  005037  001206                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
13187  064212  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
13188  064214  032777  004000  114732  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
13189  064222  001011                          BNE     1$              ;;BR IF YES
13190  064224  005737  001230                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
13191  064230  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
13192  064232  005237  001120                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
13193  064236  023737  001206  001120          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
13194  064244  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
13195  064246  012737  000001  001120  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
13196  064254  013737  064332  001206          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
13197  064262  105237  001116          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
13198  064266  113737  001116  001226          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
13199  064274  011637  001122                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
13200  064300  011637  001124                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
13201  064304  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
13202  064310  112737  000001  001131          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
13203  064316  013777  001116  114632  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
13204  064324  013716  001122                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
13205  064330  000002                          RTI                     ;;FIXES PS
13206  064332  000012                  $MXCNT: 10.                     ;;MAX. NUMBER OF ITERATIONS
13207  064334                          $SW08TBL:
13208  064334  006776                          .WORD   TST1+2          ;;STARTING ADDRESS OF TEST 1
13209  064336  007174                          .WORD   TST2+2          ;;STARTING ADDRESS OF TEST 2
13210  064340  007360                          .WORD   TST3+2          ;;STARTING ADDRESS OF TEST 3
13211  064342  007522                          .WORD   TST4+2          ;;STARTING ADDRESS OF TEST 4
13212  064344  010544                          .WORD   TST5+2          ;;STARTING ADDRESS OF TEST 5
13213  064346  011536                          .WORD   TST6+2          ;;STARTING ADDRESS OF TEST 6
13214  064350  012776                          .WORD   TST7+2          ;;STARTING ADDRESS OF TEST 7
13215  064352  014020                          .WORD   TST10+2         ;;STARTING ADDRESS OF TEST 10
13216  064354  015012                          .WORD   TST11+2         ;;STARTING ADDRESS OF TEST 11
13217  064356  016254                          .WORD   TST12+2         ;;STARTING ADDRESS OF TEST 12
13218  064360  017274                          .WORD   TST13+2         ;;STARTING ADDRESS OF TEST 13
13219  064362  020622                          .WORD   TST14+2         ;;STARTING ADDRESS OF TEST 14
13220  064364  021642                          .WORD   TST15+2         ;;STARTING ADDRESS OF TEST 15
13221  064366  022662                          .WORD   TST16+2         ;;STARTING ADDRESS OF TEST 16
13222  064370  024372                          .WORD   TST17+2         ;;STARTING ADDRESS OF TEST 17
13223  064372  025656                          .WORD   TST20+2         ;;STARTING ADDRESS OF TEST 20
13224  064374  026764                          .WORD   TST21+2         ;;STARTING ADDRESS OF TEST 21
13225  064376  027732                          .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
13226  064400  031010                          .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
13227  064402  032076                          .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
13228  064404  033070                          .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
13229  064406  034112                          .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
13230  064410  035134                          .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
13231                          .SBTTL  ERROR HANDLER ROUTINE
```

```
13232
13233                                   ;;*****************************************************************
13234                                   ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
13235                                   ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
13236                                   ;*AND GO TO ERRTYP ON ERROR
13237                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
13238                                   ;*SW15=1        HALT ON ERROR
13239                                   ;*SW13=1        INHIBIT ERROR TYPEOUTS
13240                                   ;*SW10=1        BELL ON ERROR
13241                                   ;*SW09=1        LOOP ON ERROR
13242                                   ;*CALL
13243                                   ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
13244
13245  064412                           $ERROR:
13246  064412  104410                           CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
13247  064414  105237  001117           7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
13248  064420  001775                           BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
13249  064422  013777  001116  114526           MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
13250  064430  032777  002000  114516           BIT     #BIT10,@SWR     ;;BELL ON ERROR?
13251  064436  001402                           BEQ     1$              ;;NO - SKIP
13252  064440  104401  001212                   TYPE    ,$BELL          ;;RING BELL
13253  064444  005237  001126           1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
13254  064450  011637  001132                   MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
13255  064454  162737  000002  001132           SUB     #2,$ERRPC
13256  064462  117737  114444  001130           MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
13257  064470  032777  020000  114456           BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
13258  064476  001004                           BNE     20$             ;;SKIP TYPEOUTS
13259  064500  004737  036456                   JSR     PC,ERRTYP       ;;GO TO USER ERROR ROUTINE
13260  064504  104401  001217                   TYPE    ,$CRLF
13261  064510                           20$:
13262  064510  122737  000001  001242           CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
13263  064516  001007                           BNE     2$              ;;NO,SKIP APT ERROR REPORT
13264  064520  113737  001130  064532           MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
13265  064526  004737  066454                   JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
13266  064532     000           21$:    .BYTE   0
13267  064533     000                   .BYTE   0
13268  064534  000777           22$:    BR      22$             ;;APT ERROR LOOP
13269  064536  005777  114412           2$:     TST     @SWR            ;;HALT ON ERROR
13270  064542  100002                           BPL     3$              ;;SKIP IF CONTINUE
13271  064544  000000                           HALT                    ;;HALT ON ERROR!
13272  064546  104410                           CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
13273  064550  032777  001000  114376   3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
13274  064556  001402                           BEQ     4$              ;;BR IF NO
13275  064560  013716  001124                   MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
13276  064564  005737  001210           4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
13277  064570  001402                           BEQ     5$              ;;BR IF NONE
13278  064572  013716  001210                   MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
13279  064576                           5$:
13280  064576  022737  036436  000042           CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
13281  064604  001001                           BNE     6$              ;;BRANCH IF NO
13282  064606  000000                           HALT                    ;;YES
13283  064610                           6$:
13284  064610  000002                           RTI                     ;;RETURN
13285                                   .SBTTL  TTY INPUT ROUTINE
13286
13287                                   ;;*****************************************************************
```

```
13288                                   .ENABL  LSB
13289  064612  000000                   $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
13290  064614  000000                   $TKQIN: .WORD   0               ;;INPUT POINTER
13291  064616  000000                   $TKQOUT: .WORD  0               ;;OUTPUT POINTER
13292  064620  000001                   $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
13293          064621                   $TKQEND=.
13294          064622                   .EVEN
13295
13296                                   ;*TK INITIALIZE ROUTINE
13297                                   ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
13298                                   ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
13299                                   ;
13300                                   ;*CALL:
13301                                   ;*      JSR     PC,$TKINT
13302                                   ;*      RETURN
13303                                   ;
13304  064622  005037  064612           $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
13305  064626  012737  064620  064614           MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
13306  064634  013737  064614  064616           MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
13307  064642  012737  064672  000060           MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
13308  064650  012737  000200  000062           MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
13309  064656  005777  114300                    TST     @$TKB           ;;CLEAR DONE FLAG
13310  064662  012777  000100  114270           MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
13311  064670  000207                           RTS     PC              ;;RETURN TO CALLER
13312
13313                                   ;*TK SERVICE ROUTINE
13314                                   ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
13315                                   ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
13316                                   ;*IT IN THE QUEUE.
13317                                   ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
13318                                   ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (SHUT)
13319                                   ;
13320  064672  117746  114264           $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
13321  064676  042716  177600                   BIC     #↑C177,(SP)     ;;STRIP THE JUNK
13322  064702  021627  000003                   CMP     (SP),#3         ;;IS IT A CONTROL C?
13323  064706  001007                           BNE     1$              ;;BRANCH IF NO
13324  064710  104401  066006                   TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (↑C)
13325  064714  004737  064622                   JSR     PC,$TKINT       ;;INIT THE KEYBOARD
13326  064720  005726                           TST     (SP)+           ;;CLEAN UP STACK
13327  064722  000137  062644                   JMP     SHUT            ;;CONTROL C RESTART
13328  064726  021627  000007           1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
13329  064732  001004                           BNE     2$              ;;BRANCH IF NO
13330  064734  022737  000176  001154           CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
13331  064742  001500                           BEQ     6$              ;;GO TO SWR CHANGE
13332
13333  064744                           2$:
13334  064744  022737  000001  064612           CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
13335  064752  001004                           BNE     3$              ;;BRANCH IF NO
13336  064754  104401  001212                   TYPE    ,$BELL          ;;RING THE TTY BELL
13337  064760  005726                           TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
13338  064762  000451                           BR      5$              ;;EXIT
13339  064764  021627  000023           3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
13340  064770  001021                           BNE     32$             ;;BRANCH IF NO
13341  064772  005077  114162                   CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
13342  064776  005726                           TST     (SP)+           ;;CLEAN CHAR OFF STACK
13343  065000  105777  114154           31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
```

```
13344  065004  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
13345  065006  117746  114150                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
13346  065012  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
13347  065016  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
13348  065022  001366                          BNE     31$             ;;BRANCH IF NO
13349  065024  012777  000100  114126          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
13350  065032  000002                          RTI                     ;;RETURN
13351  065034  005237  064612          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
13352  065040  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
13353  065044  002405                          BLT     4$              ;;BRANCH IF YES
13354  065046  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
13355  065052  003002                          BGT     4$              ;;BRANCH IF YES
13356  065054  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
13357  065060  112677  177530          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
13358  065064  005237  064614                  INC     $TKQIN          ;;UPDATE THE POINTER
13359  065070  023727  064614  064621          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
13360  065076  001003                          BNE     5$              ;;BRANCH IF NO
13361  065100  012737  064620  064614          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
13362  065106  000002                  5$:     RTI                     ;;RETURN
13363
13364                                  ;;**************************************************************
13365                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
13366                                  ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
13367                                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
13368                                  ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
13369  065110  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
13370  065116  001124                          BNE     15$             ;;EXIT IF NOT
13371  065120  105777  114034                  TSTB    @$TKS           ;;IS A CHAR WAITING?
13372  065124  100121                          BPL     15$             ;;IF NOT, EXIT
13373  065126  117746  114030                  MOVB    @$TKB,-(SP)     ;;YES
13374  065132  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
13375  065136  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
13376  065142  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
13377                                                                  ;;AND EXIT
13378
13379                                  ;;**************************************************************
13380                                  ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
13381                                  ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
13382                                  ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
13383  065144  123727  001150  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
13384  065152  001674                          BEQ     2$              ;;BRANCH IF YES
13385  065154  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
13386  065156  004737  064622                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
13387  065162  005077  113772                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
13388  065166  112737  000001  001151          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
13389
13390  065174  104401  066020                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
13391  065200  104401  066025          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
13392  065204  013746  000176                  MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
13393  065210  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
13394  065212  104401  066036                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
13395  065216  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
13396  065220  005046                          CLR     -(SP)           ;;THE NEW SWR
13397  065222  105777  113732          7$:     TSTB    @$TKS           ;;CHAR THERE?
13398  065226  100375                          BPL     7$              ;;IF NOT TRY AGAIN
13399
```

```
13400  065230  117746  113726          MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
13401  065234  042716  177600          BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
13402
13403  065240  021627  000003          CMP     (SP),#3         ;;IS IT A CONTROL-C?
13404  065244  001015                  BNE     9$              ;;BRANCH IF NOT
13405  065246  104401  066006          TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
13406  065252  062706  000006          ADD     #6,SP           ;;CLEAN UP STACK
13407  065256  123727  001151  000001  CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
13408  065264  001003                  BNE     8$              ;;BRANCH IF NO
13409  065266  012777  000100  113664  MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
13410  065274  000137  062644          8$:     JMP     SHUT    ;;CONTROL-C RESTART
13411
13412
13413  065300  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
13414  065304  001005                  BNE     10$             ;;BRANCH IF NOT
13415  065306  104401  066013          TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
13416  065312  062706  000006          20$:    ADD     #6,SP   ;;IGNORE PREVIOUS INPUT
13417  065316  000737                  BR      19$             ;;LET'S TRY IT AGAIN
13418
13419
13420  065320  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
13421  065324  001022                  BNE     16$             ;;BRANCH IF NO
13422  065326  005766  000004          TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
13423  065332  001403                  BEQ     11$             ;;BRANCH IF YES
13424  065334  016677  000002  113612  MOV     2(SP),@SWR      ;;SAVE NEW SWR
13425  065342  062706  000006          11$:    ADD     #6,SP   ;;CLEAR UP STACK
13426  065346  104401  001217          14$:    TYPE    ,$CRLF  ;;ECHO <CR> AND <LF>
13427  065352  123727  001151  000001  CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
13428  065360  001003                  BNE     15$             ;;BRANCH IF NOT
13429  065362  012777  000100  113570  MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
13430  065370  000002                  15$:    RTI             ;;RETURN
13431  065372  004737  063746          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
13432  065376  021627  000060          CMP     (SP),#60        ;;CHAR < 0?
13433  065402  002420                  BLT     18$             ;;BRANCH IF YES
13434  065404  021627  000067          CMP     (SP),#67        ;;CHAR > 7?
13435  065410  003015                  BGT     18$             ;;BRANCH IF YES
13436  065412  042726  000060          BIC     #60,(SP)+       ;;STRIP-OFF ASCII
13437  065416  005766  000002          TST     2(SP)           ;;IS THIS THE FIRST CHAR
13438  065422  001403                  BEQ     17$             ;;BRANCH IF YES
13439  065424  006316                  ASL     (SP)            ;;NO, SHIFT PRESENT
13440  065426  006316                  ASL     (SP)            ;;    CHAR OVER TO MAKE
13441  065430  006316                  ASL     (SP)            ;;    ROOM FOR NEW ONE.
13442  065432  005266  000002          17$:    INC     2(SP)   ;;KEEP COUNT OF CHAR
13443  065436  056616  177776          BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
13444  065442  000667                  BR      7$              ;;GET THE NEXT ONE
13445  065444  104401  001216          18$:    TYPE    ,$QUES  ;;TYPE ?<CR><LF>
13446  065450  000720                  BR      20$             ;;SIMULATE CONTROL-U
13447                                  .DSABL  LSB
13448
13449
13450                                  ;;*****************************************************************
13451                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
13452                                  ;*CALL:
13453                                  ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
13454                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
13455                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
```

```
                                        ;
065452  011646                          $RDCHR: MOV     (SP),-(SP)          ;;PUSH DOWN THE PC AND
065454  016666  000004  000002                  MOV     4(SP),2(SP)         ;;THE PS
065462  005066  000004                          CLR     4(SP)               ;;GET READY FOR A CHARACTER
065466  005046                                  CLR     -(SP)               ;;PUT NEW PS ON STACK
065470  012746  065476                          MOV     #64$,-(SP)          ;;PUT NEW PC ON  STACK
065474  000002                                  RTI                         ;;POP NEW PC AND PS
065476                                  64$:
065476  005737  064612                  1$:     TST     $TKCNT              ;;WAIT ON A CHARACTER
065502  001775                                  BEQ     1$
065504  005337  064612                          DEC     $TKCNT              ;;DECREMENT THE COUNTER
065510  117766  177102  000004                  MOVB    @$TKQOUT,4(SP)      ;;GET ONE CHARACTER
065516  005237  064616                          INC     $TKQOUT             ;;UPDATE THE POINTER
065522  023727  064616  064621                  CMP     $TKQOUT,#$TKQEND    ;;DID IT GO OFF OF THE END?
065530  001003                                  BNE     2$                  ;;BRANCH IF NO
065532  012737  064620  064616                  MOV     #$TKQSRT,$TKQOUT    ;;RESET THE POINTER
065540  000002                          2$:     RTI                         ;;RETURN
                                        ;;*******************************************************************
                                        ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                        ;*CALL:
                                        ;*      RDLIN                       ;;INPUT A STRING FROM THE TTY
                                        ;*      RETURN HERE                 ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                        ;*                                  ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

065542  010346                          $RDLIN: MOV     R3,-(SP)            ;;SAVE R3
065544  005046                                  CLR     -(SP)               ;;CLEAR THE RUBOUT KEY
065546  012703  065776                  1$:     MOV     #$TTYIN,R3          ;;GET ADDRESS
065552  022703  066006                  2$:     CMP     #$TTYIN+8.,R3       ;;BUFFER FULL?
065556  101456                                  BLOS    4$                  ;;BR IF YES
065560  104411                                  RDCHR                       ;;GO READ ONE CHARACTER FROM THE TTY
065562  112613                                  MOVB    (SP)+,(R3)          ;;GET CHARACTER
065564  122713  000177                  10$:    CMPB    #177,(R3)           ;;IS IT A RUBOUT
065570  001022                                  BNE     5$                  ;;BR IF NO
065572  005716                                  TST     (SP)                ;;IS THIS THE FIRST RUBOUT?
065574  001007                                  BNE     6$                  ;;BR IF NO
065576  112737  000134  065774                  MOVB    #'\,9$              ;;TYPE A BACK SLASH
065604  104401  065774                          TYPE    ,9$
065610  012716  177777                          MOV     #-1,(SP)            ;;SET THE RUBOUT KEY
065614  005303                          6$:     DEC     R3                  ;;BACKUP BY ONE
065616  020327  065776                          CMP     R3,#$TTYIN          ;;STACK EMPTY?
065622  103434                                  BLO     4$                  ;;BR IF YES
065624  111337  065774                          MOVB    (R3),9$             ;;SETUP TO TYPEOUT THE DELETED CHAR.
065630  104401  065774                          TYPE    ,9$                 ;;GO TYPE
065634  000746                                  BR      2$                  ;;GO READ ANOTHER CHAR.
065636  005716                          5$:     TST     (SP)                ;;RUBOUT KEY SET?
065640  001406                                  BEQ     7$                  ;;BR IF NO
065642  112737  000134  065774                  MOVB    #'\,9$              ;;TYPE A BACK SLASH
065650  104401  065774                          TYPE    ,9$
065654  005016                                  CLR     (SP)                ;;CLEAR THE RUBOUT KEY
065656  122713  000025                  7$:     CMPB    #25,(R3)            ;;IS CHARACTER A CTRL U?
065662  001003                                  BNE     8$                  ;;BR IF NO
065664  104401  066013                          TYPE    ,$CNTLU             ;;TYPE A CONTROL "U"
065670  000726                                  BR      1$                  ;;GO START OVER
065672  122713  000022                  8$:     CMPB    #22,(R3)            ;;IS CHARACTER A "↑R"?
065676  001011                                  BNE     3$                  ;;BRANCH IF NO
```

```
13512  065700  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
13513  065702  104401  001217                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
13514  065706  104401  065776                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
13515  065712  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
13516  065714  104401  001216          4$:     TYPE    ,$QUES          ;;TYPE A '?'
13517  065720  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
13518  065722  111337  065774          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
13519  065726  104401  065774                  TYPE    ,9$
13520  065732  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
13521  065736  001305                          BNE     2$              ;;LOOP IF NOT RETURN
13522  065740  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
13523  065744  104401  001220                  TYPE    ,$LF            ;;TYPE A LINE FEED
13524  065750  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
13525  065752  012603                          MOV     (SP)+,R3        ;;RESTORE R3
13526  065754  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
13527  065756  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
13528  065764  012766  065776  000004          MOV     #$TTYIN,4(SP)
13529  065772  000002                          RTI                     ;;RETURN
13530  065774     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
13531  065775     000                  .BYTE   0                       ;;TERMINATOR
13532  065776  000010          $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
13533  066006  041536  005015  000     $CNTLC: .ASCIZ  /↑C/<15><12>    ;;CONTROL "C"
13534  066013     136   006525  000012  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
13535  066020  043536  005015  000     $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
13536  066025     015   051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
13537  066032  036440  000040
13538  066036  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
13539  066044  020075     000
13540          066050                  .EVEN
13541                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
13542
13543                                  ;;**************************************************************
13544                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
13545                                  ;*CHANGE IT TO BINARY.
13546                                  ;*CALL:
13547                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
13548                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
13549                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
13550
13551  066050  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
13552  066052  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
13553  066060  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
13554  066062  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
13555  066064  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
13556  066066  104412          1$:     RDLIN                           ;;READ AN ASCIZ LINE
13557  066070  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
13558  066072  005001                          CLR     R1              ;;CLEAR DATA WORD
13559  066074  005002                          CLR     R2
13560  066076  112046          2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
13561  066100  001412                          BEQ     3$              ;;IF ZERO GET OUT
13562  066102  006301                          ASL     R1              ;;*2
13563  066104  006102                          ROL     R2
13564  066106  006301                          ASL     R1              ;;*4
13565  066110  006102                          ROL     R2
13566  066112  006301                          ASL     R1              ;;*8
13567  066114  006102                          ROL     R2
```

```
13568  066116  042716  177770          BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
13569  066122  062601                  ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
13570  066124  000764                  BR      2$              ;;LOOP
13571  066126  005726                  3$:     TST     (SP)+   ;;CLEAN TERMINATOR FROM STACK
13572  066130  010166  000012          MOV     R1,12(SP)       ;;SAVE THE RESULT
13573  066134  010237  066150          MOV     R2,$HIOCT
13574  066140  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
13575  066142  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
13576  066144  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
13577  066146  000002                  RTI                     ;;RETURN
13578  066150  000000          $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
13579                          .SBTTL  TRAP DECODER
13580
13581                          ;;**************************************************************
13582                          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
13583                          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
13584                          ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
13585                          ;*GO TO THAT ROUTINE.
13586
13587  066152  016646  000002  $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
13588  066156  042716  000020          BIC     #20,(SP)        ;;  THE CALLER--DO NOT ALLOW
13589  066162  012746  066170          MOV     #1$,-(SP)       ;;  T-BIT TRAPS
13590  066166  000002                  RTI                     ;;SET THE NEW STATUS
13591  066170  010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
13592  066172  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
13593  066176  005740                  TST     -(R0)           ;;BACKUP BY 2
13594  066200  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
13595  066202  006300                  ASL     R0              ;;POSITION FOR INDEXING
13596  066204  016000  066224          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
13597  066210  000200                  RTS     R0              ;;GO TO ROUTINE
13598
13599
13600                          ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
13601
13602  066212  011646          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
13603  066214  016666  000004  000002  MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
13604  066222  000002                  RTI                     ;;RESTORE THE PSW
13605
13606                          .SBTTL  TRAP TABLE
13607
13608                          ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
13609                          ;*BY THE "TRAP" INSTRUCTION.
13610
13611                          ;       ROUTINE
13612                          ;       -------
13613  066224  066212          $TRPAD: .WORD   $TRAP2
13614  066226  063534                  $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
13615  066230  063332                  $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
13616  066232  063306                  $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
13617  066234  063346                  $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
13618  066236  063062                  $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
13619  066240  063006                  $TYPBN  ;;CALL=TYPBN    TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER
13620
13621  066242  065200                  $GTSWR  ;;CALL=GTSWR    TRAP+7(104407)  GET SOFT-SWR SETTING
13622
13623  066244  065110                  $CKSWR  ;;CALL=CKSWR    TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
```

```
                                        $RDCHR  ;;CALL=RDCHR    TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
                                        $RDLIN  ;;CALL=RDLIN    TRAP+12(104412) TTY TYPEIN STRING ROUTINE
                                        $RDOCT  ;;CALL=RDOCT    TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
                                        $SAVREG ;;CALL=SAVREG   TRAP+14(104414) SAVE R0-R5 ROUTINE
                                        $RESREG ;;CALL=RESREG   TRAP+15(104415) RESTORE R0-R5 ROUTINE
                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;*****************************************************************
                                ;POWER DOWN ROUTINE
066260  012737  066420  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
066266  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
066274  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
066276  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
066300  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
066302  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
066304  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
066306  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
066310  017746  112640                MOV     @$WR,-(SP)      ;;PUSH @$WR ON STACK
066314  010637  066424                MOV     SP,$$AVR6       ;;SAVE SP
066320  012737  066332  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
066326  000000                        HALT
066330  000776                        BR      .-2             ;;HANG UP

                                ;;*****************************************************************
                                ;POWER UP ROUTINE
066332  012737  066420  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
066340  013706  066424                MOV     $SAVR6,SP       ;;GET SP
066344  005037  066424                CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
066350  005237  066424        1$:     INC     $SAVR6          ;;WAIT FOR THE INC
066354  001375                        BNE     1$              ;;OF  WORD
066356  012677  112572                MOV     (SP)+,@$WR      ;;POP STACK INTO @$WR
066362  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
066364  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
066366  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
066370  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
066372  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
066374  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
066376  012737  066260  000024        MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
066404  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
066412  104401                        TYPE                    ;REPORT THE POWER FAILURE
066414  066426          $PWRMG: .WORD   $POWER                  ;;POWER FAIL MESSAGE POINTER
066416  000002                        RTI
066420  000000          $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
066422  000776                        BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
066424  000000          $SAVR6: 0                               ;;PUT THE SP HERE
066426  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
066434  000122
                                        .EVEN
                                .SBTTL  APT COMMUNICATIONS ROUTINE

                                ;;*****************************************************************
066436  112737  000001  066702  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
066444  112737  000001  066700  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
066452  000403                        BR      $ATYC
066454  112737  000001  066702  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
066462                          $ATYC:
```

```
13680  066462  010046                      MOV     R0,-(SP)        ;;PUSH R0 ON STACK
13681  066464  010146                      MOV     R1,-(SP)        ;;PUSH R1 ON STACK
13682  066466  105737  066700              TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
13683  066472  001450                      BEQ     5$              ;;IF NOT:  BR
13684  066474  122737  000001  001242      CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
13685  066502  001031                      BNE     3$              ;;IF NOT:  BR
13686  066504  132737  000100  001243      BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
13687  066512  001425                      BEQ     3$              ;;IF NOT:  BR
13688  066514  017600  000004              MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
13689  066520  062766  000002  000004      ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
13690  066526  005737  001222      1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
13691  066532  001375                      BNE     1$              ;;IF NOT:  WAIT
13692  066534  010037  001236              MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
13693  066540  105720          2$:         TSTB    (R0)+           ;;FIND END OF MESSAGE
13694  066542  001376                      BNE     2$
13695  066544  163700  001236              SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
13696  066550  006200                      ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
13697  066552  010037  001240              MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
13698  066556  012737  000004  001222      MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
13699  066564  000413                      BR      5$
13700  066566  017637  000004  066612  3$: MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
13701  066574  062766  000002  000004      ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
13702  066602  013746  177776              MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
13703  066606  004737  063534              JSR     PC,$TYPE        ;;CALL TYPE MACRO
13704  066612  000000          4$:         .WORD   0
13705  066614                  5$:
13706  066614  105737  066702      10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
13707  066620  001416                      BEQ     12$             ;;IF NOT:  BR
13708  066622  005737  001242              TST     $ENV            ;;RUNNING UNDER APT?
13709  066626  001413                      BEQ     12$             ;;IF NOT:  BR
13710  066630  005737  001222      11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
13711  066634  001375                      BNE     11$             ;;IF NOT:  WAIT
13712  066636  017637  000004  001224      MOV     @4(SP),$FATAL   ;;GET ERROR #
13713  066644  062766  000002  000004      ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
13714  066652  005237  001222              INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
13715  066656  105037  066702      12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
13716  066662  105037  066701              CLRB    $LFLG           ;;CLEAR LOG FLAG
13717  066666  105037  066700              CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
13718  066672  012601                      MOV     (SP)+,R1        ;;POP STACK INTO R1
13719  066674  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
13720  066676  000207                      RTS     PC              ;;RETURN
13721  066700     000  $MFLG:   .BYTE   0       ;;MESSG. FLAG
13722  066701     000  $LFLG:   .BYTE   0       ;;LOG FLAG
13723  066702     000  $FFLG:   .BYTE   0       ;;FATAL FLAG
13724          066704              .EVEN
13725          000200  APTSIZE=200
13726          000001  APTENV=001
13727          000100  APTSPOOL=100
13728          000040  APTCSUP=040
13729
13730                  .NLIST  BEX
```

```
                                .SBTTL  CONSOLE MESSAGES

066704                          SCTMSG:
066704  005015  040503  047116  .ASCII  <CR><LF>@CANNOT RECOVER THE BAD SECTOR FILES FROM LAST @<CR><LF>
066766  051124  041501  020113  .ASCIZ  @TRACK FOR THIS DEVICE@
067014     051                  CLSPRN: .ASCII  @)@
067015     075     000          EQUALS: .ASCIZ  @=@
067017     015  025012     000  PROMPT: .ASCIZ  <CR><LF>@*@
067023     077     000          QSTMRK: .ASCIZ  @?@
067025                          HELPQST:
067025     015  052012  050131  .ASCIZ  <CR><LF>@TYPE HELP TEXT (Y OR N)??@
067061                          UBUSQST:
067061     015  041412  040510  .ASCIZ  <CR><LF>@CHANGE RM03 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N) ??@
067154  005015  051525  020105  CNSL00: .ASCIZ  <CR><LF>@USE SAME DEVICES (Y OR N) ??@
067213     015  051012  030115  CNSL01: .ASCIZ  <CR><LF>@RM03 BUS ADDRESS (@
067240  005015  047105  051124  CNSL02: .ASCII  <CR><LF>@ENTRY NOT IN I/O PAGE@
067267     015  040412  042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE >160000@
067321     015  051012  030115  CNSL03: .ASCIZ  <CR><LF>@RM03 VECTOR ADDRESS (@
067351     015  042412  052116  CNSL04: .ASCII  <CR><LF>@ENTRY OUT OF RANGE@
067375     015  040412  042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE <1000@
067425     015  051012  030115  CNSL05: .ASCIZ  <CR><LF>@RM03 INTERRUPT PRIORITY (@
067461     015  042412  052116  CNSL06: .ASCIZ  <CR><LF>@ENTRY OUT OF RANGE@
067506                          CNSL07:
067506  005015  054524  042520  .ASCII  <CR><LF>@TYPE (A) TO TEST ALL DEVICES, OR TYPE DEVICE@
067564  047040  046525  042502  .ASCII  @ NUMBER(S)@
067576  005015  042524  046522  .ASCIZ  <CR><LF>@TERMINATE INPUT WITH CARRIAGE RETURN@

        067646                  .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.

067646                  FNCDTB:                                 ;FUNCTION CODE TABLE
```

FUNCTION CODE TABLE

```
067646  020000          .WORD   OPI                                     ;NOP
067650  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (2)
067652  132000          .WORD   ATA!OPI!IVC!IAE                         ;SEEK
067654  130000          .WORD   ATA!OPI!IVC                             ;RECALIBRATE
067656  020000          .WORD   OPI                                     ;DRIVE CLEAR
067660  030000          .WORD   OPI!IVC                                 ;RELEASE
067662  130000          .WORD   OPI!ATA!IVC                             ;OFFSET
067664  130000          .WORD   OPI!ATA!IVC                             ;RETURN TO CENTERLINE
067666  020000          .WORD   OPI                                     ;READ IN PRESET
067670  020000          .WORD   OPI                                     ;PACK ACKNOWLEDGE
067672  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (24)
067674  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (26)
067676  132000          .WORD   ATA!OPI!IVC!IAE                         ;SEARCH
067700  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (32)
067702  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (34)
067704  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (36)
067706  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (40)
067710  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (42)
067712  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (44)
067714  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (46)
067716  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK DATA
067720  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK HEADER AND DATA
067722  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (54)
067724  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (56)
067726  037200          .WORD   OPI!IVC!WLE!IAE!AOE!HCE                 ;WRITE DATA
067730  037000          .WORD   OPI!IVC!WLE!IAE!AOE                     ;WRITE HEADER AND DATA
067732  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (64)
067734  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (66)
067736  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH                 ;READ DATA
067740  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH                 ;READ HEADER AND DATA
067742  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (74)
067744  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (76)
```

```
                        .SBTTL  ATTENTION (ATA) TABLE

067746  001     ATNTBL: .BYTE   1.
067747  002             .BYTE   2.
067750  004             .BYTE   4.
067751  010             .BYTE   8.
067752  020             .BYTE   16.
067753  040             .BYTE   32.
067754  100             .BYTE   64.
067755  200             .BYTE   128.
```

```
                        .SBTTL  DATA PATTERN TABLE

067756                  RGDTPT:
067756                  MIXED:
067756  000000                  .WORD   0.
067760  000001                  .WORD   1.
067762  000003                  .WORD   3.
067764  000007                  .WORD   7.
067766  000017                  .WORD   15.
067770  000037                  .WORD   31.
067772  000077                  .WORD   63.
067774  000177                  .WORD   127.
067776  000377                  .WORD   255.
070000  000777                  .WORD   511.
070002  001777                  .WORD   1023.
070004  003777                  .WORD   2047.
070006  007777                  .WORD   4095.
070010  017777                  .WORD   8191.
070012  037777                  .WORD   16383.
070014  077777                  .WORD   32767.
070016  177777          ONES:   .WORD   65535.
070020  177777                  .WORD   65535.
070022  077777                  .WORD   32767.
070024  037777                  .WORD   16383.
070026  017777                  .WORD   8191.
070030  007777                  .WORD   4095.
070032  003777                  .WORD   2047.
070034  001777                  .WORD   1023.
070036  000777                  .WORD   511.
070040  000377                  .WORD   255.
070042  000177                  .WORD   127.
070044  000077                  .WORD   63.
070046  000037                  .WORD   31.
070050  000017                  .WORD   15.
070052  000007                  .WORD   7.
070054  000003                  .WORD   3.
070056  000001                  .WORD   1.
070060  000000          ZEROS:  .WORD   0.
070062  000000                  .WORD   0.
070064  000001                  .WORD   1.
070066  000002                  .WORD   2.
070070  000004                  .WORD   4.
070072  000010                  .WORD   8.
070074  000020                  .WORD   16.
070076  000040                  .WORD   32.
070100  000100                  .WORD   64.
070102  000200                  .WORD   128.
070104  000400                  .WORD   256.
070106  001000                  .WORD   512.
070110  002000                  .WORD   1024.
070112  004000                  .WORD   2048.
070114  010000                  .WORD   4096.
070116  020000                  .WORD   8192.
070120  040000                  .WORD   16384.
070122  100000                  .WORD   32768.
070124  100000                  .WORD   32768.
```

```
070126  040000                          .WORD   16384.
070130  020000                          .WORD   8192.
070132  010000                          .WORD   4096.
070134  004000                          .WORD   2048.
070136  002000                          .WORD   1024.
070140  001000                          .WORD   512.
070142  000400                          .WORD   256.
070144  000200                          .WORD   128.
070146  000100                          .WORD   64.
070150  000040                          .WORD   32.
070152  000020                          .WORD   16.
070154  000010                          .WORD   8.
070156  000004                          .WORD   4.
070160  000002                          .WORD   2.
070162  000001                          .WORD   1.
070164  000000                          .WORD   0.
070166  177777                          .WORD   65535.
070170  177776                          .WORD   65534.
070172  177774                          .WORD   65532.
070174  177770                          .WORD   65528.
070176  177760                          .WORD   65520.
070200  177740                          .WORD   65504.
070202  177700                          .WORD   65472.
070204  177600                          .WORD   65408.
070206  177400                          .WORD   65280.
070210  177000                          .WORD   65024.
070212  176000                          .WORD   64512.
070214  174000                          .WORD   63488.
070216  170000                          .WORD   61440.
070220  160000                          .WORD   57344.
070222  140000                          .WORD   49152.
070224  100000                          .WORD   32768.
070226  000000                          .WORD   0.
070230  000000                          .WORD   0.
070232  100000                          .WORD   32768.
070234  140000                          .WORD   49152.
070236  160000                          .WORD   57344.
070240  170000                          .WORD   61440.
070242  174000                          .WORD   63488.
070244  176000                          .WORD   64512.
070246  177000                          .WORD   65024.
070250  177400                          .WORD   65280.
070252  177600                          .WORD   65408.
070254  177700                          .WORD   65472.
070256  177740                          .WORD   65504.
070260  177760                          .WORD   65520.
070262  177770                          .WORD   65528.
070264  177774                          .WORD   65532.
070266  177776                          .WORD   65534.
070270  177777                          .WORD   65535.
070272  125252                  EARLY:  .WORD   43690.
070274  152525                          .WORD   43690./2
070276  125252                          .WORD   43690.
070300  177777                          .WORD   65535.
070302  177776                          .WORD   65534.
070304  177775                          .WORD   65533.
```

```
070306  177773                  .WORD   65531.
070310  177767                  .WORD   65527.
070312  177757                  .WORD   65519.
070314  177737                  .WORD   65503.
070316  177677                  .WORD   65471.
070320  177577                  .WORD   65407.
070322  177377                  .WORD   65279.
070324  176777                  .WORD   65023.
070326  175777                  .WORD   64511.
070330  173777                  .WORD   63487.
070332  167777                  .WORD   61439.
070334  157777                  .WORD   57343.
070336  137777                  .WORD   49151.
070340  077777                  .WORD   32767.
070342  077777                  .WORD   32767.
070344  137777                  .WORD   49151.
070346  157777                  .WORD   57343.
070350  167777                  .WORD   61439.
070352  173777                  .WORD   63487.
070354  175777                  .WORD   64511.
070356  176777                  .WORD   65023.
070360  177377                  .WORD   65279.
070362  177577                  .WORD   65407.
070364  177677                  .WORD   65471.
070366  177737                  .WORD   65503.
070370  177757                  .WORD   65519.
070372  177767                  .WORD   65527.
070374  177773                  .WORD   65531.
070376  177775                  .WORD   65533.
070400  177776                  .WORD   65534.
070402  177777                  .WORD   65535.
070404                  ENRGDT:
```

```
                                .SBTTL  ERROR MESSAGE TABLE

070404  075000  000000          EMT1:   .WORD   EMS1,0
070410  075047  075064  000000  EMT2:   .WORD   EMS2,EMS3,0
070416  075047  075127  000000  EMT3:   .WORD   EMS2,EMS4,0
070424  075172  075222  000000  EMT4:   .WORD   EMS5,EMS6,0
070432  075172  075334  000000  EMT5:   .WORD   EMS5,EMS10,0
070440  102027  077215  000000  EMT6:   .WORD   EMS167,EMS64,0
070446  077763  102054  000000  EMT7:   .WORD   EMS110,EMS170,0
070454  075267  000000          EMT10:  .WORD   EMS7,0
070460  075334  000000          EMT11:  .WORD   EMS10,0
070464  075376  075407  000000  EMT12:  .WORD   EMS11,EMS12,0
070472  075450  075461  075472  EMT13:  .WORD   EMS13,EMS14,EMS15,EMS16,0
070504  075544  077215  000000  EMT14:  .WORD   EMS17,EMS64,0
070512  075376  075627  000000  EMT15:  .WORD   EMS11,EMS21,0
070520  075376  075652  076001  EMT16:  .WORD   EMS11,EMS22,EMS27,0
070530  075376  075666  076012  EMT17:  .WORD   EMS11,EMS23,EMS30,0
070540  075376  075714  076012  EMT20:  .WORD   EMS11,EMS24,EMS30,0
070550  075376  075743  076001  EMT21:  .WORD   EMS11,EMS25,EMS27,0
070560  075376  075760  076001  EMT22:  .WORD   EMS11,EMS26,EMS27,0
070570  075376  076022  076012  EMT23:  .WORD   EMS11,EMS31,EMS30,0
070600  075376  076051  076012  EMT24:  .WORD   EMS11,EMS32,EMS30,0
070610  075376  076100  076012  EMT25:  .WORD   EMS11,EMS33,EMS30,0
070620  075376  076126  076012  EMT26:  .WORD   EMS11,EMS34,EMS30,0
070630  075376  076177  076012  EMT27:  .WORD   EMS11,EMS35,EMS30,0
070640  075376  076226  076012  EMT30:  .WORD   EMS11,EMS36,EMS30,0
070650  075376  076255  076012  EMT31:  .WORD   EMS11,EMS37,EMS30,0
070660  075376  076303  076012  EMT32:  .WORD   EMS11,EMS40,EMS30,0
070670  075376  076332  076012  EMT33:  .WORD   EMS11,EMS41,EMS30,0
070700  075376  076360  076012  EMT34:  .WORD   EMS11,EMS42,EMS30,0
070710  075376  076407  076012  EMT35:  .WORD   EMS11,EMS43,EMS30,0
070720  075376  076436  076012  EMT36:  .WORD   EMS11,EMS44,EMS30,0
070730  075376  076511  076012  EMT37:  .WORD   EMS11,EMS45,EMS30,0
070740  077301  075604  000000  EMT40:  .WORD   EMS66,EMS20,0
070746  077467  101175  077475  EMT41:  .WORD   EMS75,EMS141,EMS76,0
070756  101266  101276  077423  EMT42:  .WORD   EMS144,EMS145,EMS72,EMS76,0
070770  076603  076721  077475  EMT43:  .WORD   EMS47,EMS53,EMS76,0
071000  077526  076721  077475  EMT44:  .WORD   EMS77,EMS53,EMS76,0
071010  077554  076721  077475  EMT45:  .WORD   EMS100,EMS53,EMS76,0
071020  077602  076721  077475  EMT46:  .WORD   EMS101,EMS53,EMS76,0
071030  077233  077215  000000  EMT47:  .WORD   EMS65,EMS64,0
071036  075450  075472  077167  EMT50:  .WORD   EMS13,EMS15,EMS63,0
071046  075376  076554  076012  EMT51:  .WORD   EMS11,EMS46,EMS30,EMS67,0
071060  076603  076721  077351  EMT52:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS140,EMS141,0
071076  076603  076721  077351  EMT53:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS141,EMS164,0
071114  076632  076721  077351  EMT54:  .WORD   EMS50,EMS53,EMS67,0
071124  101223  101240  076721  EMT55:  .WORD   EMS142,EMS143,EMS53,EMS67,0
071136  076661  077423  077351  EMT56:  .WORD   EMS51,EMS72,EMS67,EMS115,EMS50,EMS70,0
071154  102027  077433  077351  EMT57:  .WORD   EMS167,EMS73,EMS67,0
071164  077004  077351  100163  EMT60:  .WORD   EMS56,EMS67,EMS115,EMS150,EMS152,EMS70,0
071202  077410  077004  077351  EMT61:  .WORD   EMS71,EMS56,EMS67,EMS115,EMS150,EMS152,EMS72,0
071222  101154  077351  100163  EMT62:  .WORD   EMS140,EMS67,EMS115,EMS47,EMS70,0
071236  000000                  EMT63:  .WORD
071240  101361  101424  077376  EMT64:  .WORD   EMS150,EMS152,EMS70,EMS115,EMS56,EMS73,EMS67,0
071260  076707  102454  102635  EMT65:  .WORD   EMS52,EMS205,EMS214,EMS206,EMS115,EMS51,EMS72,0
071300  101766  077657  077423  EMT66:  .WORD   EMS165,EMS103,EMS72,EMS124,0
```

```
071312  101766  077657  077423  EMT67:  .WORD   EMS165,EMS103,EMS72,EMS171,0
071324  076554  075604  101661  EMT70:  .WORD   EMS46,EMS20,EMS163,0
071334  077410  077602  101661  EMT71:  .WORD   EMS71,EMS101,EMS163,0
071344  076603  077423  101661  EMT72:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS140,EMS141,0
071362  076603  077423  101661  EMT73:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS141,EMS72,0
071400  077004  076721  101661  EMT74:  .WORD   EMS56,EMS53,EMS163,0
071410  077410  077004  101661  EMT75:  .WORD   EMS71,EMS56,EMS163,EMS115,EMS150,EMS152,EMS72,0
071430  076632  076721  101661  EMT76:  .WORD   EMS50,EMS53,EMS163,0
071440  101223  101240  076721  EMT77:  .WORD   EMS142,EMS143,EMS53,EMS163,0
071452  077467  101175  101661  EMT100: .WORD   EMS75,EMS141,EMS163,EMS115,EMS47,EMS70,0
071470  077467  101361  101661  EMT101: .WORD   EMS75,EMS150,EMS163,EMS115,EMS56,EMS73,0
071506  102027  077433  101661  EMT102: .WORD   EMS167,EMS73,EMS163,0
071516  101344  101400  077450  EMT103: .WORD   EMS147,EMS151,EMS74,EMS163,0
071530  076661  077450  101661  EMT104: .WORD   EMS51,EMS74,EMS163,EMS115,EMS50,EMS70,0
071546  101766  077554  101661  EMT105: .WORD   EMS165,EMS100,EMS163,0
071556  101766  077526  101661  EMT106: .WORD   EMS165,EMS77,EMS163,0
071566  101266  101276  076721  EMT107: .WORD   EMS144,EMS145,EMS53,EMS163,EMS115,EMS143,EMS70,0
071606  077763  100023  100170  EMT110: .WORD   EMS110,EMS112,EMS116,EMS111,0
071620  100107  075127  000000  EMT111: .WORD   EMS113,EMS4,0
071626  100107  075064  000000  EMT112: .WORD   EMS113,EMS3,0
071634  077763  100163  100170  EMT113: .WORD   EMS110,EMS115,EMS116,EMS117,EMS114,0
071650  100107  100242  000000  EMT114: .WORD   EMS113,EMS120,0
071656  100257  100717  100743  EMT115: .WORD   EMS121,EMS132,EMS133,0
071666  100322  100717  100743  EMT116: .WORD   EMS122,EMS132,EMS133,0
071676  100357  100717  100743  EMT117: .WORD   EMS123,EMS132,EMS133,0
071706  100422  100717  100743  EMT120: .WORD   EMS124,EMS132,EMS133,
071716  100454  100717  100743  EMT121: .WORD   EMS125,EMS132,EMS133,0
071726  100517  100717  100743  EMT122: .WORD   EMS126,EMS132,EMS133,0
071736  101117  100717  100743  EMT123: .WORD   EMS137,EMS132,EMS133,0
071746  100621  100717  100743  EMT124: .WORD   EMS130,EMS132,EMS133,0
071756  100657  100717  100743  EMT125: .WORD   EMS131,EMS132,EMS133,0
071766  100257  100717  100766  EMT126: .WORD   EMS121,EMS132,EMS134,EMS123,0
072000  100322  100717  100766  EMT127: .WORD   EMS122,EMS132,EMS134,EMS123,0
072012  100357  100717  100766  EMT130: .WORD   EMS123,EMS132,EMS134,EMS123,0
072024  100422  100717  100766  EMT131: .WORD   EMS124,EMS132,EMS134,EMS123,0
072036  100454  100717  100766  EMT132: .WORD   EMS125,EMS132,EMS134,EMS123,0
072050  100517  100717  100766  EMT133: .WORD   EMS126,EMS132,EMS134,EMS123,0
072062  101117  100717  100766  EMT134: .WORD   EMS137,EMS132,EMS134,EMS123,0
072074  100621  100717  100766  EMT135: .WORD   EMS130,EMS132,EMS134,EMS123,0
072106  100657  100717  100766  EMT136: .WORD   EMS131,EMS132,EMS134,EMS123,0
072120  100257  100717  101030  EMT137: .WORD   EMS121,EMS132,EMS135,0
072130  100357  100717  101030  EMT140: .WORD   EMS123,EMS132,EMS135,0
072140  100257  100717  101072  EMT141: .WORD   EMS121,EMS132,EMS136,0
072150  101117  100717  101072  EMT142: .WORD   EMS137,EMS132,EMS136,0
072160  100422  100717  101072  EMT143: .WORD   EMS124,EMS132,EMS136,0
072170  100454  100717  101072  EMT144: .WORD   EMS125,EMS132,EMS136,0
072200  100517  100717  101072  EMT145: .WORD   EMS126,EMS132,EMS136,0
072210  100657  100717  101072  EMT146: .WORD   EMS131,EMS132,EMS136,0
072220  101626  100717  101072  EMT147: .WORD   EMS162,EMS132,EMS136,0
072230  100621  100717  101072  EMT150: .WORD   EMS130,EMS132,EMS136,0
072240  101154  100163  101175  EMT151: .WORD   EMS140,EMS115,EMS141,EMS70,0
072252  101223  100163  101240  EMT152: .WORD   EMS142,EMS115,EMS143,EMS72,0
072264  101266  101276  101325  EMT153: .WORD   EMS144,EMS145,EMS146,EMS115,EMS143,EMS72,0
072302  101266  101276  075604  EMT154: .WORD   EMS144,EMS145,EMS20,EMS115,EMS143,EMS70,0
072320  101361  101424  101472  EMT155: .WORD   EMS150,EMS152,EMS154,EMS153,0
072332  101344  101400  101472  EMT156: .WORD   EMS147,EMS151,EMS154,EMS155,0
```

```
072344  101344  101400  101526  EMT157: .WORD   EMS147,EMS151,EMS156,EMS157,0
072356  101576  101526  101561  EMT160: .WORD   EMS161,EMS156,EMS160,0
072366  075743  076001  101526  EMT161: .WORD   EMS25,EMS27,EMS156,EMS160,0
072400  075760  076001  101526  EMT162: .WORD   EMS26,EMS27,EMS156,EMS160,0
072412  076603  101661  101154  EMT163: .WORD   EMS47,EMS163,EMS140,0
072422  076603  075604  000000  EMT164: .WORD   EMS47,EMS20,0
072430  077004  075604  000000  EMT165: .WORD   EMS56,EMS20,0
072436  076554  075604  000000  EMT166: .WORD   EMS46,EMS20,0
072444  103155  075604  000000  EMT167: .WORD   EMS224,EMS20,0
072452  102027  101472  102002  EMT170: .WORD   EMS167,EMS154,EMS166,0
072462  102027  101472  101544  EMT171: .WORD   EMS167,EMS154,EMS157,0
072472  102027  101472  101506  EMT172: .WORD   EMS167,EMS154,EMS155,0
072502  102103  100717  100743  EMT173: .WORD   EMS171,EMS132,EMS133,0
072512  102103  100717  100766  EMT174: .WORD   EMS171,EMS132,EMS134,EMS123,0
072524  100107  102256  000000  EMT175: .WORD   EMS113,EMS177,0
072532  102300  102315  000000  EMT176: .WORD   EMS200,EMS201,0
072540  102413  101175  076012  EMT177: .WORD   EMS203,EMS141,EMS30,EMS202,0
072552  102413  076661  076012  EMT200: .WORD   EMS203,EMS51,EMS30,EMS202,0
072564  102413  102426  076012  EMT201: .WORD   EMS203,EMS204,EMS30,EMS202,0
072576  102413  076632  076012  EMT202: .WORD   EMS203,EMS50,EMS30,EMS202,0
072610  102413  101240  076012  EMT203: .WORD   EMS203,EMS143,EMS30,EMS202,0
072622  101361  077450  102363  EMT204: .WORD   EMS150,EMS74,EMS202,EMS115,EMS152,EMS72,0
072640  076632  077657  077423  EMT205: .WORD   EMS50,EMS103,EMS72,0
072650  077666  077433  102363  EMT206: .WORD   EMS104,EMS73,EMS202,0
072660  077467  101175  100163  EMT207: .WORD   EMS75,EMS141,EMS115,EMS140,0
072672  077467  101361  000000  EMT210: .WORD   EMS75,EMS150,0
072700  076661  077423  100163  EMT211: .WORD   EMS51,EMS72,EMS115,EMS50,EMS70,0
072714  101223  076721  100163  EMT212: .WORD   EMS142,EMS53,EMS115,EMS143,EMS72,0
072730  076632  077657  076721  EMT213: .WORD   EMS50,EMS103,EMS53,0
072740  076707  102454  102002  EMT214: .WORD   EMS52,EMS205,EMS166,EMS206,EMS115,EMS51,EMS72,0
072760  076707  100220  077004  EMT215: .WORD   EMS52,EMS117,EMS56,EMS163,0
072772  077004  076012  077215  EMT216: .WORD   EMS56,EMS30,EMS64,0
073002  076554  076012  077215  EMT217: .WORD   EMS46,EMS30,EMS64,0
073012  076022  076012  077215  EMT220: .WORD   EMS31,EMS30,EMS64,0
073022  076603  076012  077215  EMT221: .WORD   EMS47,EMS30,EMS64,0
073032  076707  100220  077526  EMT222: .WORD   EMS52,EMS117,EMS77,0
073042  076707  100220  077167  EMT223: .WORD   EMS52,EMS117,EMS63,0
073052  000000                  EMT224: .WORD
073054  000000                  EMT225: .WORD
073056  000000                  EMT226: .WORD
073060  000000                  EMT227: .WORD
073062  000000                  EMT230: .WORD
073064  000000                  EMT231: .WORD
073066  000000                  EMT232: .WORD
073070  000000                  EMT233: .WORD
073072  000000                  EMT234: .WORD
073074  000000                  EMT235: .WORD
073076  000000                  EMT236: .WORD
073100  000000                  EMT237: .WORD
073102  000000                  EMT240: .WORD
073104  000000                  EMT241: .WORD
073106  000000                  EMT242: .WORD
073110  000000                  EMT243: .WORD
073112  000000                  EMT244: .WORD
073114  000000                  EMT245: .WORD
073116  102027  100717  102513  EMT246: .WORD   EMS167,EMS132,EMS207,0
```

```
073126  102027  100717  102540  EMT247: .WORD   EMS167,EMS132,EMS210,EMS125,0
073140  102027  102551  102540  EMT250: .WORD   EMS167,EMS211,EMS210,EMS207,EMS206,0
073154  102567  102612  000000  EMT251: .WORD   EMS212,EMS213,0
073162  101766  102567  000000  EMT252: .WORD   EMS165,EMS212,0
073170  101344  101400  101526  EMT253: .WORD   EMS147,EMS151,EMS156,EMS210,EMS26,EMS27,0
073206  102413  077602  076012  EMT254: .WORD   EMS203,EMS101,EMS30,0
073216  102413  102[illegible]7  076012  EMT255: .WORD   EMS203,EMS167,EMS30,0
073226  102413  077554  076012  EMT256: .WORD   EMS203,EMS100,EMS30,0
073236  075376  076554  076012  EMT257: .WORD   EMS11,EMS46,EMS30,EMS102,0
073250  076707  100220  077004  EMT260: .WORD   EMS52,EMS117,EMS46,EMS102,0
073262  076707  102454  102757  EMT261: .WORD   EMS52,EMS115,EMS20,EMS206,EMS115,EMS51,EMS72,0
073302  077666  077433  102363  EMT262: .WORD   EMS104,EMS73,EMS202,0
073312  076632  076721  077630  EMT263: .WORD   EMS50,EMS53,EMS102,0
073322  077004  076721  077630  EMT264: .WORD   EMS56,EMS53,EMS102,EMS115,EMS150,EMS152,EMS70,0
073342  077410  077004  077630  EMT265: .WORD   EMS71,EMS56,EMS102,EMS115,EMS150,EMS152,EMS72,0
073362  101223  101240  076721  EMT266: .WORD   EMS142,EMS143,EMS53,EMS102,0
073374  076632  101325  100163  EMT267: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,0
073412  076603  076721  077630  EMT270: .WORD   EMS47,EMS53,EMS102,EMS115,EMS140,0
073426  076603  076721  077630  EMT271: .WORD   EMS47,EMS53,EMS102,EMS115,EMS141,EMS72,0
073444  077467  101175  077630  EMT272: .WORD   EMS75,EMS141,EMS102,EMS115,EMS47,EMS73,0
073462  076661  077450  077630  EMT273: .WORD   EMS51,EMS74,EMS102,EMS115,EMS50,EMS70,0
073500  077167  076721  077061  EMT274: .WORD   EMS63,EMS53,EMS57,EMS115,EMS41,EMS146,0
073516  100170  076721  076332  EMT275: .WORD   EMS116,EMS53,EMS41,EMS57,0
073530  076603  076721  077061  EMT276: .WORD   EMS47,EMS53,EMS57,EMS115,EMS140,0
073544  076603  076721  077061  EMT277: .WORD   EMS47,EMS53,EMS57,EMS115,EMS141,EMS72,0
073562  077004  076721  077061  EMT300: .WORD   EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS70,0
073602  077410  077004  076721  EMT301: .WORD   EMS71,EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS72,0
073624  101766  077554  077657  EMT302: .WORD   EMS165,EMS100,EMS103,EMS57,0
073636  101766  077602  077657  EMT303: .WORD   EMS165,EMS101,EMS103,EMS57,0
073650  101766  077526  077657  EMT304: .WORD   EMS166,EMS77,EMS103,EMS57,0
073662  076554  076012  077215  EMT305: .WORD   EMS46,EMS30,EMS64,EMS57,0
073674  076632  076721  077061  EMT306: .WORD   EMS50,EMS53,EMS57,0
073704  076632  101325  100163  EMT307: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,EMS57,0
073724  101223  101240  076721  EMT310: .WORD   EMS142,EMS143,EMS53,EMS57,0
073736  077666  077657  076721  EMT311: .WORD   EMS104,EMS103,EMS53,EMS57,0
073750  077715  077657  076721  EMT312: .WORD   EMS105,EMS103,EMS53,EMS57,0
073762  101266  101276  077657  EMT313: .WORD   EMS144,EMS145,EMS103,EMS57,EMS115,EMS143,EMS70,0
074002  076360  077657  076721  EMT314: .WORD   EMS42,EMS103,EMS53,EMS57,0
074014  076022  077657  076721  EMT315: .WORD   EMS31,EMS103,EMS53,EMS57,0
074026  077410  076022  077657  EMT316: .WORD   EMS71,EMS31,EMS103,EMS57,0
074040  076407  077657  077061  EMT317: .WORD   EMS43,EMS103,EMS57,0
074050  076511  077657  077061  EMT320: .WORD   EMS45,EMS103,EMS57,0
074060  076436  077657  077061  EMT321: .WORD   EMS44,EMS103,EMS57,0
074070  077744  075604  000000  EMT322: .WORD   EMS106,EMS20,0
074076  076226  077657  077061  EMT323: .WORD   EMS36,EMS103,EMS57,0
074106  102156  076226  077657  EMT324: .WORD   EMS173,EMS36,EMS103,EMS57,0
074120  102136  076226  077657  EMT325: .WORD   EMS172,EMS36,EMS103,EMS57,0
074132  075450  102173  075472  EMT326: .WORD   EMS13,EMS174,EMS15,EMS35,EMS53,EMS175,0
074150  101344  101400  077450  EMT327: .WORD   EMS147,EMS151,EMS74,EMS175,0
074162  077301  076721  102201  EMT330: .WORD   EMS66,EMS53,EMS175,0
074172  076100  077657  076721  EMT331: .WORD   EMS33,EMS103,EMS53,EMS175,0
074204  076303  077657  076721  EMT332: .WORD   EMS40,EMS103,EMS53,EMS57,0
074216  076661  077450  077061  EMT333: .WORD   EMS51,EMS74,EMS57,EMS115,EMS50,EMS70,0
074234  077467  101175  077061  EMT334: .WORD   EMS75,EMS141,EMS57,EMS115,EMS47,EMS73,0
074252  077467  101361  101424  EMT335: .WORD   EMS75,EMS150,EMS152,EMS57,EMS115,EMS56,EMS73,0
074272  077107  077122  077151  EMT336: .WORD   EMS60,EMS61,EMS62,0
```

```
074302  100170  100220  076126  EMT337: .WORD   EMS116,EMS117,EMS34,0
074312  076126  076721  076733  EMT340: .WORD   EMS34,EMS53,EMS54,EMS111,0
074324  076755  076126  000000  EMT341: .WORD   EMS55,EMS34,0
074332  076707  100220  077004  EMT342: .WORD   EMS52,EMS112,EMS56,EMS57,0
074344  076707  100220  076511  EMT343: .WORD   EMS52,EMS117,EMS45,EMS57,0
074356  076707  100220  076436  EMT344: .WORD   EMS52,EMS117,EMS44,EMS57,0
074370  076707  100220  102777  EMT345: .WORD   EMS52,EMS117,EMS221,0
074400  077744  075604  100163  EMT346: .WORD   EMS106,EMS20,EMS115,EMS223,EMS72,0
074414  076707  102454  103047  EMT347: .WORD   EMS52,EMS205,EMS222,EMS206,0
074426  077467  101361  077630  EMT350: .WORD   EMS75,EMS150,EMS102,EMS115,EMS56,EMS73,0
074444  102027  077433  077630  EMT351: .WORD   EMS167,EMS73,EMS102,0
074454  102653  000000          EMT352: .WORD   EMS215,0
074460  102724  102413  102674  EMT353: .WORD   EMS217,EMS203,EMS216,0
074470  102777  075604  000000  EMT354: .WORD   EMS221,EMS20,0
```

```
074476  103227  104033  104110  EHT1:   .WORD   EH1,STSH1,STSH2,STSH4,0
074510  104033  104110  104235  EHT2:   .WORD   STSH1,STSH2,STSH4,0

074520  103246  000000          EHT110: .WORD   EH110,0
074524  103255  000000          EHT111: .WORD   EH111,0

074530  103274  000000          EHT114: .WORD   EH114,0
074534  103323  104033  104110  EHT223: .WORD   EH223,STSH1,STSH2,STSH4,0
074546  103351  104033  104110  EHT256: .WORD   EH256,STSH1,STSH2,STSH4,0

074560  103426  104033  104110  EHT336: .WORD   EH336,STSH1,STSH2,STSH4,0
074572  103465  104033  104110  EHT337: .WORD   EH337,STSH1,STSH2,STSH4,0
074604  103624  104033  104110  EHT344: .WORD   EH344,STSH1,STSH2,STSH4,0

074616  103764  000000          EHT353: .WORD   EH353,0
```

```
074622  104274  104370  104406  EDT1:   .WORD   ED1,STSD1,STSD2,STSD4
074632  104370  104406  104440  EDT2:   .WORD   STSD1,STSD2,STSD4

074640  104302                  EDT110: .WORD   ED110
074642  104306                  EDT111: .WORD   ED111

074644  104314                  EDT114: .WORD   ED114
074646  104324  104370  104406  EDT223: .WORD   ED223,STSD1,STSD2,STSD4

074656  104334  104370  104406  EDT336: .WORD   ED336,STSD1,STSD2,STSD4
074666  104346  104370  104406  EDT337: .WORD   ED337,STSD1,STSD2,STSD4
074676  104346  104370  104406  EDT344: .WORD   ED337,STSD1,STSD2,STSD4,0

074710  104360                  EDT353: .WORD   ED353
```

```
074712  104453  104471  104471  EFT1:   .WORD   EF111,STSF,STSF,STSF
074722  104471  104471  104471  EFT2:   .WORD   STSF,STSF,STSF

074730  104452                  EFT110: .WORD   EF110
074732  104453                  EFT111: .WORD   EF111

074734  104455                  EFT114: .WORD   EF114
074736  104455  104471  104471  EFT223: .WORD   EF114,STSF,STSF,STSF

074746  104460  104471  104471  EFT336: .WORD   EF336,STSF,STSF,STSF
074756  104460  104471  104471  EFT337: .WORD   EF336,STSF,STSF,STSF
074766  104460  104471  104471  EFT344: .WORD   EF336,STSF,STSF,STSF

074776  104455                  EFT353: .WORD   EF114
```

```
                        .SBTTL  ERROR MESSAGE STRINGS

075000 051127 047117 020107 EMS1:   .ASCIZ  @WRONG UNIT SELECTED (RMCS2, BITS 0-2) @
075047    104 053105 041511 EMS2:   .ASCIZ  @DEVICE WENT @
075064 047125 053101 044501 EMS3:   .ASCIZ  @UNAVAILABLE "DVA" (RMCS1, BIT 11) @
075127    116 047117 054105 EMS4:   .ASCIZ  @NONEXISTENT "NED" (RMCS2, BIT 12) @
075172 047503 046515 047101 EMS5:   .ASCIZ  @COMMAND NOT COMPLETED @
075222 047503 052116 047522 EMS6:   .ASCIZ  @CONTROLLER NOT READY (RMCS1, BIT 7) @
075267    104 044522 042526 EMS7:   .ASCIZ  @DRIVE NOT READY "DRY" (RMDS, BIT 7) @
075334 047507 047040 052117 EMS10:  .ASCIZ  @GO NOT RESET "GO" (RMCS1, BIT 0) @
075376 047111 040526 044514 EMS11:  .ASCIZ  @INVALID @
075407    106 047125 052103 EMS12:  .ASCIZ  @FUNCTION CODE (RMCS1, BITS 1-5) @
075450 040515 051523 052502 EMS13:  .ASCIZ  @MASSBUS @
075461    103 047117 051124 EMS14:  .ASCIZ  @CONTROL @
075472 052502 020123 040520 EMS15:  .ASCIZ  @BUS PARITY ERROR @
075514 046442 050103 021105 EMS16:  .ASCIZ  @"MCPE" (RMCS1, BIT 13) @
075544 051124 047101 043123 EMS17:  .ASCIZ  @TRANSFER ERROR (RMCS1, BIT 14) @
075604 044123 052517 042114 EMS20:  .ASCIZ  @SHOULD NOT BE SET @
075627    127 051117 020104 EMS21:  .ASCIZ  @WORD COUNT (RMWC) @
075652 052502 020123 051050 EMS22:  .ASCIZ  @BUS (RMBA) @
075666 046042 052102 020042 EMS23:  .ASCIZ  @"LBT" (RMDS, BIT 10) @
075714 040442 042517 020042 EMS24:  .ASCIZ  @"AOE" (RMER1, BIT 09) @
075743    104 051511 020113 EMS25:  .ASCIZ  @DISK (RMDA) @
075760 054503 044514 042116 EMS26:  .ASCIZ  @CYLINDER (RMDC) @
076001    101 042104 042522 EMS27:  .ASCIZ  @ADDRESS @
076012 052123 052101 051525 EMS30:  .ASCIZ  @STATUS @
076022 053442 042514 020042 EMS31:  .ASCIZ  @"WLE" (RMER1, BIT 11) @
076051    042 050125 021105 EMS32:  .ASCIZ  @"UPE" (RMCS2, BIT 13) @
076100 053442 043103 020042 EMS33:  .ASCIZ  @"WCF" (RMER1, BIT 5) @
076126 051127 052111 020105 EMS34:  .ASCIZ  @WRITE CHECK ERROR-"WCE" (RMCS2, BIT 14) @
076177    042 042115 042520 EMS35:  .ASCIZ  @"MDPE" (RMCS2, BIT 8) @
076226 042042 045503 020042 EMS36:  .ASCIZ  @"DCK" (RMER1, BIT 15) @
076255    042 041505 021110 EMS37:  .ASCIZ  @"ECH" (RMER1, BIT 6) @
076303    042 046104 021124 EMS40:  .ASCIZ  @"DLT" (RMCS2, BIT 15) @
076332 046442 043130 020042 EMS41:  .ASCIZ  @"MXF" (RMCS2, BIT 9) @
076360 042042 042524 020042 EMS42:  .ASCIZ  @"DTE" (RMER1, BIT 12) @
076407    042 041510 041522 EMS43:  .ASCIZ  @"HCRC" (RMER1, BIT 8) @
076436 042510 042101 051105 EMS44:  .ASCIZ  @HEADER COMPARE ERROR "HCE" (RMER1, BIT 7) @
076511    106 051117 040515 EMS45:  .ASCIZ  @FORMAT ERROR "FER" (RMER1, BIT 4) @
076554 044442 042501 020042 EMS46:  .ASCIZ  @"IAE" (RMER1, BIT 10) @
076603    042 050117 021111 EMS47:  .ASCIZ  @"OPI" (RMER1, BIT 13) @
076632 051442 044513 020042 EMS50:  .ASCIZ  @"SKI" (RMER2, BIT 14) @
076661    042 044520 021120 EMS51:  .ASCIZ  @"PIP" (RMDS, BIT 13) @
076707    124 042510 051040 EMS52:  .ASCIZ  @THE RM03 @
076721    104 052105 041505 EMS53:  .ASCIZ  @DETECTED @
076733    101 020124 047101 EMS54:  .ASCIZ  @AT AN UNEXPECTED @
076755    111 041516 051117 EMS55:  .ASCIZ  @INCORRECT DATA DURING @
077004 047111 040526 044514 EMS56:  .ASCIZ  @INVALID COMMAND ERROR "IVC" (RMER2, BIT 12) @
077061    104 051125 047111 EMS57:  .ASCIZ  @DURING DATA TRANSFER @
077107    104 052101 020101 EMS60:  .ASCIZ  @DATA READ @
077122 047504 051505 047040 EMS61:  .ASCIZ  @DOES NOT COMPARE WITH @
077151    104 052101 020101 EMS62:  .ASCIZ  @DATA WRITTEN @
077167    042 040520 021122 EMS63:  .ASCIZ  @"PAR" (RMER1, BIT 3) @
077215    111 020123 047111 EMS64:  .ASCIZ  @IS INCORRECT @
077233    103 046517 047520 EMS65:  .ASCIZ  @COMPOSITE ERROR "ERR" (RMDS, BIT 14) @
077301    104 052101 020101 EMS66:  .ASCIZ  @DATA PARITY ERROR "DPE" (RMER2, BIT 3) @
```

```
077351   104  051125  047111  EMS67:  .ASCIZ  @DURING SEEK COMMAND @
077376 051511  051040  051505  EMS70:  .ASCIZ  @IS RESET @
077410 051105  047522  042516  EMS71:  .ASCIZ  @ERRONEOUS @
077423   111  020123  042523  EMS72:  .ASCIZ  @IS SET @
077433   104  042111  047040  EMS73:  .ASCIZ  @DID NOT SET @
077450 044504  020104  047516  EMS74:  .ASCIZ  @DID NOT RESET @
077467   114  051517  020124  EMS75:  .ASCIZ  @LOST @
077475   104  051125  047111  EMS76:  .ASCIZ  @DURING PACK ACK COMMAND @
077526 051042  051115  020042  EMS77:  .ASCIZ  @"RMR" (RMER1, BIT 2) @
077554 044442  051114  020042  EMS100: .ASCIZ  @"ILR" (RMER1, BIT 1) @
077602 044442  043114  020042  EMS101: .ASCIZ  @"ILF" (RMER1, BIT 0) @
077630 052504  044522  043516  EMS102: .ASCIZ  @DURING SEARCH COMMAND @
077657   105  051122  051117  EMS103: .ASCIZ  @ERROR @
077666 046042  041502  020042  EMS104: .ASCIZ  @"LBC" (RMER2, BIT 10) @
077715   042  051514  021103  EMS105: .ASCIZ  @"LSC" (RMER2, BIT 11) @
077744 042510  042101  051105  EMS106: .ASCIZ  @HEADER ERRORS @
077763   102  051525  052040  EMS110: .ASCIZ  @BUS TIMEOUT (04 TRAP) @
100012 042101  051104  051505  EMS111: .ASCIZ  @ADDRESS @
100023   127  042510  020116  EMS112: .ASCII  @WHEN READING/WRITING RH REGISTERS @
100065   101  020124  044124          .ASCIZ  @AT THE FOLLOWING @
100107   124  042510  051440  EMS113: .ASCIZ  @THE SELECTED DEVICE IS @
100137   116  047117  054105  EMS114: .ASCIZ  @NONEXISTENT DEVICE @
100163   040  006455  000012  EMS115: .ASCIZ  @ -@<CR><LF>
100170 044124  020105  040515  EMS116: .ASCIZ  @THE MASSBUS CONTROLLER @
100220 040506  046111  042105  EMS117: .ASCIZ  @FAILED TO DETECT @
100242 047516  020124  047101  EMS120: .ASCIZ  @NOT AN RM03 @
100257   103  047117  051124  EMS121: .ASCIZ  @CONTROL STATUS REGISTER 1, RMCS1, @
100322 052502  020123  042101  EMS122: .ASCIZ  @BUS ADDRESS REGISTER, RMBA, @
100357   103  047117  051124  EMS123: .ASCIZ  @CONTROL STATUS REGISTER 2, RMCS2, @
100422 051105  047522  020122  EMS124: .ASCIZ  @ERROR REGISTER 1, RMER1, @
100454 052101  042524  052116  EMS125: .ASCIZ  @ATTENTION SUMMARY REGISTER, RMAS, @
100517   115  044501  052116  EMS126: .ASCIZ  @MAINTENANCE REGISTER #1, RMMR #1, @
100562 041505  020103  047520  EMS127: .ASCIZ  @ECC POSITION REGISTER, RMEC1, @
100621   105  041503  050040  EMS130: .ASCIZ  @ECC PATTERN REGISTER, RMEC2, @
100657   115  044501  052116  EMS131: .ASCIZ  @MAINTENANCE REGISTER 2, RMMR2, @
100717   116  052117  044440  EMS132: .ASCIZ  @NOT INITIALIZED BY @
100743   125  044516  052502  EMS133: .ASCIZ  @UNIBUS INITIALIZE @
100766 047503  052116  047522  EMS134: .ASCIZ  @CONTROLLER CLEAR, I.E.  BIT 5 OF @
101030 044122  030461  042440  EMS135: .ASCIZ  @RH11 ERROR CLEAR (RMCS1, BIT 14) @
101072 051104  053111  020105  EMS136: .ASCIZ  @DRIVE CLEAR COMMAND @
101117   104  044522  042526  EMS137: .ASCIZ  @DRIVE STATUS REGISTER, RMDS @
101154 042515  044504  046525  EMS140: .ASCIZ  @MEDIUM OFF LINE @
101175   042  047515  021114  EMS141: .ASCIZ  @"MOL" (RMDS, BIT 12) @
101223   104  044522  042526  EMS142: .ASCIZ  @DRIVE FAULT @
101240 042042  041526  020042  EMS143: .ASCIZ  @"DVC" (RMER2, BIT 7) @
101266 047125  040523  042506  EMS144: .ASCIZ  @UNSAFE @
101276 052442  051516  020042  EMS145: .ASCIZ  @"UNS" (RMER1, BIT 14) @
101325   123  047510  046125  EMS146: .ASCIZ  @SHOULD BE SET @
101344 043117  051506  052105  EMS147: .ASCIZ  @OFFSET MODE @
101361   040  047526  052514  EMS150: .ASCIZ  @ VOLUME VALID @
101400 047442  021115  024040  EMS151: .ASCIZ  @"OM" (RMDS, BIT 0) @
101424 053042  021126  024040  EMS152: .ASCIZ  @"VV" (RMDS, BIT 6) @
101450 040520  045503  040440  EMS153: .ASCIZ  @PACK ACK COMMAND @
101472 047516  020124  042523  EMS154: .ASCIZ  @NOT SET BY @
101506 043117  051506  052105  EMS155: .ASCIZ  @OFFSET COMMAND @
101526 047516  020124  042522  EMS156: .ASCIZ  @NOT RESET BY @
```

```
101544 052122 020103 047503 EMS157: .ASCIZ @RTC COMMAND @
101561    122 050111 041440 EMS160: .ASCIZ @RIP COMMAND @
101576 043117 051506 052105 EMS161: .ASCIZ @OFFSET REGISTER (RMOF) @
101626 051105 047522 020122 EMS162: .ASCIZ @ERROR REGISTER #2, RMER2, @
101661    104 051125 047111 EMS163: .ASCIZ @DURING RECALIBRATE @
101705    111 020123 047111 EMS164: .ASCII @IS INTERMITTENT OR DRIVE DIDNT DROP ON @
101754 054503 044514 042116       .ASCIZ @CYLINDER @
101766 047125 054105 042520 EMS165: .ASCIZ @UNEXPECTED @
102002 042522 040503 044514 EMS166: .ASCIZ @RECALIBRATE COMMAND @
102027    042 052101 021101 EMS167: .ASCIZ @"ATA" (RMDS, BIT15) @
102054 044127 047105 051040 EMS170: .ASCIZ @WHEN READING REGISTER @
102103    105 051122 051117 EMS171: .ASCIZ @ERROR REGISTER #2, RMER2, @
102136 047516 051116 041505 EMS172: .ASCIZ @NONRECOVERABLE @
102156 042522 047503 042526 EMS173: .ASCIZ @RECOVERABLE @
102173    104 052101 020101 EMS174: .ASCIZ @DATA @
102201    104 051125 047111 EMS175: .ASCIZ @DURING WRITE COMMAND @
102227    042 050117 021105 EMS176: .ASCIZ @"OPE" (RMER2, BIT 13) @
102256 047111 053440 044522 EMS177: .ASCIZ @IN WRITE PROTECT @
102300 040503 020116 047516 EMS200: .ASCIZ @CAN NOT SET @
102315    104 040511 047107 EMS201: .ASCIZ @DIAGNOSTIC MODE "DMD" (RMMR1, BIT 0) @
102363    104 051125 047111 EMS202: .ASCIZ @DURING DIAGNOSTIC MODE @
102413    111 041516 051117 EMS203: .ASCIZ @INCORRECT @
102426 053442 046122 020042 EMS204: .ASCIZ @"WRL" (RMDS, BIT 11) @
102454 054105 041505 052125 EMS205: .ASCIZ @EXECUTED @
102466 044527 044124 041440 EMS206: .ASCIZ @WITH COMP ERROR SET @
102513    042 047507 020042 EMS207: .ASCIZ @"GO" (RMCS1, BIT 0) @
102540 051127 052111 047111 EMS210: .ASCIZ @WRITING @
102551    127 051501 051040 EMS211: .ASCIZ @WAS RESET BY @
102567    120 047522 051107 EMS212: .ASCIZ @PROGRAM INTERRUPT @
102612 040527 020123 047516 EMS213: .ASCIZ @WAS NOT GENERATED @
102635    123 042505 020113 EMS214: .ASCIZ @SEEK COMMAND @
102653    120 047522 051107 EMS215: .ASCIZ @PROGRAM TIMEOUT @
102674 052504 044522 043516 EMS216: .ASCIZ @DURING LOOK AHEAD TEST @
102724 047514 045517 040440 EMS217: .ASCIZ @LOOK AHEAD REGISTER,RMLA, @
102757    123 040505 041522 EMS220: .ASCIZ @SEARCH COMMAND @
102777    102 042101 051440 EMS221: .ASCIZ @BAD SECTOR ERROR "BSE" (RMER2, BIT 15) @
103047    101 042040 052101 EMS222: .ASCIZ @A DATA TRANSFER COMMAND @
103100 042510 042101 051105 EMS223: .ASCIZ @HEADER COMPARE INHIBIT "HCI" (RMOF, BIT 10) @
103155    116 047117 054105 EMS224: .ASCIZ @NONEXISTENT MEMORY "NEM" (RMCS2, BIT 11) @
```

```
103227    105  050130  052103  EH1:    .ASCIZ  @EXPCTD  RECEVD@
103246 052502  040523  051104  EH110:  .ASCIZ  @BUSADR@
103255    040  046522  051503  EH111:  .ASCIZ  @ RMCS2   RMCS1@

103274 042522  042503  042126  EH114:  .ASCIZ  @RECEVD  SNGPRT  DULPRT@
103323    105  050130  052103  EH223:  .ASCIZ  @EXPCTD  RECEVD   DATA@
103351    105  050130  052103  EH256:  .ASCII  @EXPCTD  RECEVD  RGSTR@<CR><LF>
103400 052123  052101  051525          .ASCIZ  @STATUS  STATUS  INDEX@

103426 042107  042101  051522  EH336:  .ASCIZ  @GDADRS  GDDATA  BDADRS  BDDATA@
103465    122  041515  031123  EH337:  .ASCII  @RMCS2   STATUS  FAILING   DATA@<CR><LF>
103525    137  057537  057537          .ASCII  @+++++++++++++++  +++++++++++++++@<CR><LF>
103565    105  050130  052103          .ASCIZ  @EXPCTD  RECEVD    BIT    ADRESS@

103624 046522  051105  020061  EH344:  .ASCII  @RMER1   STATUS  HEADER  FAILING@<CR><LF>
103665    137  057537  057537          .ASCII  @+++++++++++++++   WORD     BIT@<CR><LF>
103724 054105  041520  042124          .ASCIZ  @EXPCTD  RECEVD  NUMBER  POSITON@
103764 054105  041520  042124  EH353:  .ASCII  @EXPCTD  RECEVD@<CR><LF>
104004 051040  046115  020101          .ASCIZ  @ RMLA    RMLA    RMOF @

104033    040  046522  051503  STSH1:  .ASCII  @ RMCS1  RMCS2   RMDS    RMER1   RMER2@
104100 020040  051040  040515          .ASCIZ  @   RMAS@
104110 051040  053515  020103  STSH2:  .ASCII  @ RMWC    RMBA    RMDA    RMOF    RMDC@
104155    040  020040  051040          .ASCIZ  @    RMEC1   RMEC2@
104177    040  046522  040504  STSH3:  .ASCIZ  @ RMDA    RMDC    RMOF    RMLA@
104235    040  046522  051115  STSH4:  .ASCIZ  @ RMMR1   RMMR2   RMDT    RMSN@
```

```
                104274                          .EVEN
104274  001140  001142  000000  ED1:    .WORD   $GDDAT,$BDDAT,0
104302  001276  000000          ED110:  .WORD   $BASE,0
104306  001174  001176  000000  ED111:  .WORD   $TMP0,$TMP1,0

104314  001354  001176  001200  ED114:  .WORD   RMDTI,$TMP1,$TMP2,0
104324  001140  001142  001174  ED223:  .WORD   $GDDAT,$BDDAT,$TMP0,0

104334  001134  001140  001136  ED336:  .WORD   $GDADR,$GDDAT,$BDADR,$BDDAT,0

104346  001140  001142  001174  ED337:  .WORD   $GDDAT,$BDDAT,$TMP0,$TMP1,0
104360  001140  001142  001430  ED353:  .WORD   $GDDAT,$BDDAT,RMOF0,0

104370  001326  001336  001340  STSD1:  .WORD   RMCS1I,RMCS2I,RMDSI,RMER1I,RMER2I,RMASI,0
104406  001330  001332  001334  STSD2:  .WORD   RMWCI,RMBAI,RMDAI,RMOFI,RMDCI,RMEC1I
104422  001374  000000          .WORD   RMEC2I,0
104426  001334  001362  001360  STSD3:  .WORD   RMDAI,RMDCI,RMOFI,RMLAI,0
104440  001352  001366  001354  STSD4:  .WORD   RMMR1I,RMMR2I,RMDTI,RMSNI,0
```

```
104452  000                     EF110:  .BYTE   0
104453  000     000             EF111:  .BYTE   0,0
104455  000     000     000     EF114:  .BYTE   0,0,0
104460  000     000     000     EF336:  .BYTE   0,0,0,0
104464  000     000     000     EF337:  .BYTE   0,0,0,0,0

104471  000     000     000     STSF:   .BYTE   0,0,0,0,0,0,0
```

```
104500  000402
105504  177777
105506  177777
105510  000402
106514  177777
106516  177777
```

```
.EVEN
MFGFIL: .BLKW   258.
        .WORD   -1
        .WORD   -1
USRFIL:         .BLKW   258.
        .WORD   -1
        .WORD   -1
.EVEN
```

```
106520                          BUFFER:
106520  000402                  BUFONE: .BLKW   258.
107524  000402                  BUFTWO: .BLKW   258.

        106520                          =       BUFFER
106520                          HELP:
106520  005015                  .ASCII  <CR><LF>
106522  040503  052125  047511  .ASCII  @CAUTION: PARTS 2 AND 3 OF THE FUNCTIONAL TEST LEAVE BAD@<CR><LF>
106613     110  040505  042504  .ASCII  @HEADERS ON THE PACK SO BE SURE TO FORMAT THE PACK WHEN@<CR><LF>
106703     104  047117  027105  .ASCII  @DONE.@<CR><LF>
106712  046011  051511  020124  .ASCII  @       LIST OF TESTS@<CR><LF>
106732  057411  057537  057537  .ASCII  @       ++++++++++++@<CR><LF>
106752  030524  041411  047117  .ASCII  @T1     CONTROLLER ACCESS TEST@<CR><LF>
107005     124  004462  042504  .ASCII  @T2     DEVICE AVAILABLE TEST@<CR><LF>
107037     124  004463  051104  .ASCII  @T3     DRIVE TYPE TEST@<CR><LF>
107063     124  004464  051127  .ASCII  @T4     WRITE, READ ZEROS@<CR><LF>
107111     124  004465  051127  .ASCII  @T5     WRITE, WRITE CHECK ZEROS@<CR><LF>
107146  033124  053411  044522  .ASCII  @T6     WRITE, WRITE CHECK ZEROS W/ WCE ERROR@<CR><LF>
107220  033524  053411  044522  .ASCII  @T7     WRITE, READ ONES@<CR><LF>
107245     124  030061  053411  .ASCII  @T10    WRITE, WRITE CHECK ONES@<CR><LF>
107302  030524  004461  051127  .ASCII  @T11    WRITE, WRITE CHECK ONES W/ WCE ERROR@<CR><LF>
107354  030524  004462  051127  .ASCII  @T12    WRITE, WRITE CHECK MULTIPLE SECTORS@<CR><LF>
107425     124  031461  053411  .ASCII  @T13    WRITE, READ W/ IMPLIED SEEK@<CR><LF>
107466  030524  004464  051127  .ASCII  @T14    WRITE, WRITE CHECK W/ HEAD SWITCHING@<CR><LF>
107540  030524  004465  051127  .ASCII  @T15    WRITE, WRITE CHECK W/ MID-TRANSFER SEEK@<CR><LF>
107615     124  033061  053411  .ASCII  @T16    WRITE, READ W/ HCE ERROR@<CR><LF>
107653     124  033461  053411  .ASCII  @T17    WRITE, READ W/ HCI@<CR><LF>
107703     124  030062  053411  .ASCII  @T20    WRITE, READ W/ IVC ERROR@<CR><LF>
107741     124  030462  053411  .ASCII  @T21    WRITE, READ W/ ABORT@<CR><LF>
107773     124  031062  053411  .ASCII  @T22    WRITE, READ W/ BAI@<CR><LF>
110023     124  031462  053411  .ASCII  @T23    WRITE, READ EACH CURRENT LEVEL@<CR><LF>
110067     124  032062  053411  .ASCII  @T24    WRITE, WRITE CHECK W/ CURRENT LEVEL SWITCHING@<CR><LF>
110152  031124  004465  051127  .ASCII  @T25    WRITE, READ EARLY PEAK SHIFT@<CR><LF>
110214  031124  004466  051127  .ASCII  @T26    WRITE, READ MIXED PEAK SHIFT@<CR><LF>
110256  031124  004467  051127  .ASCII  @T27    WRITE, READ EACH TRACK@<CR><LF>
110312  005015                  .ASCII  <CR><LF>
110314  047411  042520  040522  .ASCII  @       OPERATIONAL SWITCH SETTINGS@<CR><LF>
110352  057411  057537  057537  .ASCII  @       ++++++++++++++++++++++++++@<CR><LF>
110410  005015                  .ASCII  <CR><LF>
110412  053523  052111  044103  .ASCII  @SWITCH                  USE@<CR><LF>
110430  026455  026455  026455  .ASCII  @------                  ----------@<CR><LF>
110466  020040  032461  004411  .ASCII  @  15                    HALT ON ERROR@<CR><LF>
110513     040  030440  004464  .ASCII  @  14                    LOOP ON TEST@<CR><LF>
110537     040  030440  004463  .ASCII  @  13                    INHIBIT ERROR TYPEOUTS@<CR><LF>
110575     040  030440  004462  .ASCII  @  12                    @<CR><LF>
110605     040  030440  004461  .ASCII  @  11                    INHIBIT ITERATIONS@<CR><LF>
110637     040  030440  004460  .ASCII  @  10                    BELL ON ERROR@<CR><LF>
110664  020040  034440  004411  .ASCII  @   9                    LOOP ON ERROR@<CR><LF>
110711     040  020040  004470  .ASCII  @   8                    LOOP ON TEST IN SWR<7:0>@<CR><LF>
110751     040  020040  004467  .ASCII  @   7                    TN128@<CR><LF>
110766  020040  033040  004411  .ASCII  @   6                    TN64@<CR><LF>
111002  020040  032440  004411  .ASCII  @   5                    TN32@<CR><LF>
111016  020040  032040  004411  .ASCII  @   4                    TN16@<CR><LF>
111032  020040  031440  004411  .ASCII  @   3                    TN8@<CR><LF>
111045     040  020040  004462  .ASCII  @   2                    TN4@<CR><LF>
```

```
111060  020040  030440  004411  .ASCII  @  1          TN2@<CR><LF>
111073     040  020040  004460  .ASCII  @  0          TN1@<CR><LF>
111106  005015                  .ASCII  <CR><LF>
111110  005015     000          .ASCIZ  <CR><LF>
        000001                  .END
```

```
ABASE = 176700          1215#  1323   1364
ACDW1 = 000000          1323   1366
ACDW2 = 000000          1323   1367
ACKSTS  053136          9269   9457   11512#
ACPUOP= 000000          1323   1338
ADDW0 = 000000          1323   1368
ADDW1 = 000000          1323   1369
ADDW10= 000000          1323
ADDW11= 000000          1323
ADDW12= 000000          1323
ADDW13= 000000          1323
ADDW14= 000000          1323
ADDW15= 000000          1323
ADDW2 = 000000          1323   1370
ADDW3 = 000000          1323   1371
ADDW4 = 000000          1323   1372
ADDW5 = 000000          1323   1373
ADDW6 = 000000          1323   1374
ADDW7 = 000000          1323   1375
ADDW8 = 000000          1323
ADDW9 = 000000          1323
ADEVCT= 000000          1323   1329
ADEVM = 000000          1323   1365
ADR   = 000001          3453#  3454   3495#  3496#  3497   3539#  3540#  3541   3573#  3574#  3575   3830#  3831#
                        3832   4078#  4079#  4080   4370#  4371#  4372   4627#  4628#  4629   4875#  4876#  4877
                        5167#  5168#  5169   5420#  5421#  5422   5736#  5737#  5738   5989#  5990#  5991   6242#
                        6243#  6244   6618#  6619#  6620   6919#  6920#  6921   7157#  7158#  7159   7369#  7370#
                        7371   7634#  7635#  7636   7898#  7899#  7900   8146#  8147#  8148   8403#  8404#  8405
                        8660#  8661#  8662
AENV  = 000000          1323   1334
AENVM = 000000          1323   1335
AFATAL= 000000          1323   1326
AMADR1= 000000          1323   1351
AMADR2= 000000          1323   1355
AMADR3= 000000          1323   1358
AMADR4= 000000          1323   1361
AMAMS1= 000000          1323   1345
AMAMS2= 000000          1323   1353
AMAMS3= 000000          1323   1356
AMAMS4= 000000          1323   1359
AMSGAD= 000000          1323   1331
AMSGLG= 000000          1323   1332
AMSGTY= 000000          1323   1325
AMTYP1= 000000          1323   1346
AMTYP2= 000000          1323   1354
AMTYP3= 000000          1323   1357
AMTYP4= 000000          1323   1360
AOE   = 001000           995#  1006   10682  10697  10712  10727  10731  10978  10982  13730
APASS = 000000          1323   1328
APE   = 100000          1194#
APRIOR= 000000          1323
APTCSU= 000040         13083  13728#
APTENV= 000001         13076  13262  13684  13726#
APTSIZ= 000200          3230  13725#
APTSPO= 000100         13078  13686  13727#
ARGS  = 000006          3586#  3604#  3626#  3634#  3637#  3640#  3648#  3668#  3692#  3700#  3703#  3706#  3714#
```

```
                3721#  3730#  3749#  3771#  3779#  3782#  3785#  3793#  3800#  3809#  3816#  3843#  3861#
                3883#  3891#  3894#  3897#  3905#  3925#  3949#  3957#  3960#  3963#  3971#  3978#  3987#
                4006#  4027#  4035#  4038#  4041#  4049#  4056#  4065#  4091#  4109#  4131#  4139#  4142#
                4145#  4153#  4173#  4197#  4205#  4208#  4211#  4219#  4226#  4235#  4257#  4270#  4286#
                4289#  4292#  4300#  4309#  4350#  4383#  4401#  4423#  4431#  4434#  4437#  4445#  4465#
                4489#  4497#  4500#  4503#  4511#  4518#  4527#  4546#  4568#  4576#  4579#  4582#  4590#
                4597#  4606#  4613#  4640#  4658#  4680#  4688#  4691#  4694#  4702#  4722#  4746#  4754#
                4757#  4760#  4768#  4775#  4784#  4803#  4824#  4832#  4835#  4838#  4846#  4853#  4862#
                4888#  4906#  4928#  4936#  4939#  4942#  4950#  4970#  4994#  5002#  5005#  5008#  5016#
                5023#  5032#  5054#  5075#  5083#  5086#  5089#  5097#  5106#  5147#  5180#  5198#  5220#
                5228#  5231#  5234#  5242#  5269#  5291#  5299#  5302#  5305#  5313#  5320#  5329#  5348#
                5369#  5377#  5380#  5383#  5391#  5398#  5407#  5433#  5451#  5473#  5481#  5484#  5487#
                5495#  5518#  5529#  5537#  5538#  5539#  5545#  5569#  5577#  5580#  5588#  5595#  5604#
                5626#  5637#  5645#  5646#  5647#  5653#  5676#  5685#  5688#  5696#  5703#  5712#  5719#
                5749#  5767#  5789#  5797#  5800#  5803#  5811#  5838#  5860#  5868#  5871#  5874#  5882#
                5889#  5898#  5917#  5938#  5946#  5949#  5952#  5960#  5967#  5976#  6002#  6020#  6042#
                6050#  6053#  6056#  6064#  6091#  6113#  6121#  6124#  6127#  6135#  6142#  6151#  6170#
                6191#  6199#  6202#  6205#  6213#  6220#  6229#  6259#  6277#  6306#  6314#  6317#  6320#
                6328#  6348#  6372#  6380#  6383#  6386#  6394#  6415#  6434#  6451#  6463#  6481#  6504#
                6512#  6515#  6518#  6526#  6546#  6565#  6582#  6594#  6633#  6651#  6680#  6688#  6691#
                6694#  6702#  6722#  6746#  6754#  6757#  6760#  6768#  6788#  6797#  6815#  6837#  6845#
                6848#  6851#  6859#  6879#  6888#  6895#  6935#  6950#  6992#  7000#  7003#  7006#  7014#
                7023#  7037#  7056#  7097#  7105#  7108#  7111#  7119#  7128#  7142#  7174#  7188#  7222#
                7223#  7229#  7243#  7246#  7254#  7262#  7281#  7313#  7315#  7323#  7338#  7341#  7349#
                7356#  7382#  7400#  7422#  7430#  7433#  7436#  7444#  7464#  7493#  7501#  7504#  7507#
                7515#  7522#  7531#  7550#  7575#  7583#  7586#  7589#  7597#  7604#  7613#  7620#  7647#
                7665#  7687#  7695#  7698#  7701#  7709#  7729#  7753#  7761#  7764#  7767#  7775#  7782#
                7791#  7810#  7833#  7841#  7844#  7847#  7855#  7862#  7871#  7878#  7911#  7929#  7951#
                7959#  7962#  7965#  7973#  7993#  8017#  8025#  8028#  8031#  8039#  8046#  8055#  8074#
                8095#  8103#  8106#  8109#  8117#  8124#  8133#  8159#  8177#  8199#  8207#  8210#  8213#
                8221#  8241#  8265#  8273#  8276#  8279#  8287#  8294#  8303#  8322#  8344#  8352#  8355#
                8358#  8366#  8373#  8382#  8389#  8416#  8434#  8456#  8464#  8467#  8470#  8478#  8498#
                8522#  8530#  8533#  8536#  8544#  8551#  8560#  8579#  8601#  8609#  8612#  8615#  8623#
                8630#  8639#  8646#  8673#  8691#  8713#  8721#  8724#  8727#  8735#  8755#  8779#  8787#
                8790#  8793#  8801#  8808#  8817#  8836#  8858#  8866#  8869#  8872#  8880#  8887#  8896#
                8903#
ASNDA   001476  1446#  9647*  9676   9722*  9723   9725*  9726*  9727   9729*  9774
ASNDC   001474  1445#  9646*  9675   9730*  9731   9733*  9773
ASWREG= 000000  1323   1336
ATA     = 100000   975#  10509  10510  10512  11244  11245  11280  11283  11775  11776  11815  11818  13730
ATESTN= 000000  1323   1327
ATNMSK= 000377  1012#  11972
ATNTBL  067746  3396   3414   13730#
AUNIT   = 000000  1323   1330
AUSWR   = 000000  1323   1337
AVECT1= 120254  1216#  1323   1362
AVECT2= 000000  1323   1363
A16     = 000400  1165#
A17     = 001000  1164#
BACK    = 000000  3586#  3591   3604#  3608   3626#  3630   3634#  3637#  3640#  3644   3648#  3652   3668#
                3673   3692#  3696   3700#  3703#  3706#  3710   3714#  3718   3721#  3725   3730#  3734
                3749#  3754   3771#  3775   3779#  3782#  3785#  3789   3793#  3797   3800#  3804   3809#
                3813   3816#  3822   3843#  3848   3861#  3865   3883#  3887   3891#  3894#  3897#  3901
                3905#  3909   3925#  3930   3949#  3953   3957#  3960#  3963#  3967   3971#  3975   3978#
                3982   3987#  3991   4006#  4011   4027#  4031   4035#  4038#  4041#  4045   4049#  4053
                4056#  4060   4065#  4069   4091#  4096   4109#  4113   4131#  4135   4139#  4142#  4145#
```

```
4149    4153#   4157    4173#   4178    4197#   4201    4205#   4208#   4211#   4215    4219#   4223
4226#   4230    4235#   4239    4257#   4262    4278#   4282    4286#   4289#   4292#   4296    4300#
4304    4309#   4313    4350#   4354    4383#   4388    4401#   4405    4423#   4427    4431#   4434#
4437#   4441    4445#   4449    4465#   4470    4489#   4493    4497#   4500#   4503#   4507    4511#
4515    4518#   4522    4527#   4531    4546#   4551    4568#   4572    4576#   4579#   4582#   4586
4590#   4594    4597#   4601    4606#   4610    4613#   4619    4640#   4645    4658#   4662    4680#
4684    4688#   4691#   4694#   4698    4702#   4706    4722#   4727    4746#   4750    4754#   4757#
4760#   4764    4768#   4772    4775#   4779    4784#   4788    4803#   4808    4824#   4828    4832#
4835#   4838#   4842    4846#   4850    4853#   4857    4862#   4866    4888#   4893    4906#   4910
4928#   4932    4936#   4939#   4942#   4946    4950#   4954    4970#   4975    4994#   4998    5002#
5005#   5008#   5012    5016#   5020    5023#   5027    5032#   5036    5054#   5059    5075#   5079
5083#   5086#   5089#   5093    5097#   5101    5106#   5110    5147#   5151    5180#   5185    5198#
5202    5220#   5224    5228#   5231#   5234#   5238    5242#   5246    5269#   5274    5291#   5295
5299#   5302#   5305#   5309    5313#   5317    5320#   5324    5329#   5333    5348#   5353    5369#
5373    5377#   5380#   5383#   5387    5391#   5395    5398#   5402    5407#   5411    5433#   5438
5451#   5455    5473#   5477    5481#   5484#   5487#   5491    5495#   5499    5518#   5523    5529#
5533    5537#   5538#   5539#   5543    5545#   5549    5569#   5573    5577#   5580#   5584    5588#
5592    5595#   5599    5604#   5608    5626#   5631    5637#   5641    5645#   5646#   5647#   5651
5653#   5657    5676#   5680    5685#   5688#   5692    5696#   5700    5703#   5707    5712#   5716
5719#   5725    5749#   5754    5767#   5771    5789#   5793    5797#   5800#   5803#   5807    5811#
5815    5838#   5843    5860#   5864    5868#   5871#   5874#   5878    5882#   5886    5889#   5893
5898#   5902    5917#   5922    5938#   5942    5946#   5949#   5952#   5956    5960#   5964    5967#
5971    5976#   5980    6002#   6007    6020#   6024    6042#   6046    6050#   6053#   6056#   6060
6064#   6068    6091#   6096    6113#   6117    6121#   6124#   6127#   6131    6135#   6139    6142#
6146    6151#   6155    6170#   6175    6191#   6195    6199#   6202#   6205#   6209    6213#   6217
6220#   6224    6229#   6233    6259#   6264    6277#   6281    6306#   6310    6314#   6317#   6320#
6324    6328#   6332    6348#   6353    6372#   6376    6380#   6383#   6386#   6390    6394#   6398
6415#   6419    6434#   6438    6451#   6455    6463#   6467    6481#   6486    6504#   6508    6512#
6515#   6518#   6522    6526#   6530    6546#   6550    6565#   6569    6582#   6586    6594#   6598
6633#   6638    6651#   6655    6680#   6684    6688#   6691#   6694#   6698    6702#   6706    6722#
6727    6746#   6750    6754#   6757#   6760#   6764    6768#   6772    6788#   6792    6797#   6801
6815#   6820    6837#   6841    6845#   6848#   6851#   6855    6859#   6863    6879#   6883    6888#
6892    6895#   6901    6935#   6950#   6955    6992#   6996    7000#   7003#   7006#   7010    7014#
7018    7023#   7027    7037#   7041    7056#   7061    7097#   7101    7105#   7108#   7111#   7115
7119#   7123    7128#   7132    7142#   7146    7174#   7188#   7193    7222#   7223#   7227    7229#
7233    7243#   7246#   7250    7254#   7258    7262#   7266    7281#   7286    7313#   7315#   7319
7323#   7327    7338#   7341#   7345    7349#   7353    7356#   7360    7382#   7387    7400#   7404
7422#   7426    7430#   7433#   7436#   7440    7444#   7448    7464#   7469    7493#   7497    7501#
7504#   7507#   7511    7515#   7519    7522#   7526    7531#   7535    7550#   7555    7575#   7579
7583#   7586#   7589#   7593    7597#   7601    7604#   7608    7613#   7617    7620#   7626    7647#
7652    7665#   7669    7687#   7691    7695#   7698#   7701#   7705    7709#   7713    7729#   7734
7753#   7757    7761#   7764#   7767#   7771    7775#   7779    7782#   7786    7791#   7795    7810#
7815    7833#   7837    7841#   7844#   7847#   7851    7855#   7859    7862#   7866    7871#   7875
7878#   7884    7911#   7916    7929#   7933    7951#   7955    7959#   7962#   7965#   7969    7973#
7977    7993#   7998    8017#   8021    8025#   8028#   8031#   8035    8039#   8043    8046#   8050
8055#   8059    8074#   8079    8095#   8099    8103#   8106#   8109#   8113    8117#   8121    8124#
8128    8133#   8137    8159#   8164    8177#   8181    8199#   8203    8207#   8210#   8213#   8217
8221#   8225    8241#   8246    8265#   8269    8273#   8276#   8279#   8283    8287#   8291    8294#
8298    8303#   8307    8322#   8327    8344#   8348    8352#   8355#   8358#   8362    8366#   8370
8373#   8377    8382#   8386    8389#   8395    8416#   8421    8434#   8438    8456#   8460    8464#
8467#   8470#   8474    8478#   8482    8498#   8503    8522#   8526    8530#   8533#   8536#   8540
8544#   8548    8551#   8555    8560#   8564    8579#   8584    8601#   8605    8609#   8612#   8615#
8619    8623#   8627    8630#   8634    8639#   8643    8646#   8652    8673#   8678    8691#   8695
8713#   8717    8721#   8724#   8727#   8731    8735#   8739    8755#   8760    8779#   8783    8787#
8790#   8793#   8797    8801#   8805    8808#   8812    8817#   8821    8836#   8841    8858#   8862
8866#   8869#   8872#   8876    8880#   8884    8887#   8891    8896#   8900    8903#   8909
```

```
BADSCT  040174          3604    3861    4109    4401    4658    4906    5198    5451    5767    6020    6277    6651    7400
                        7665    7929    8177    8434    8691    9363#
BAI    = 000010         1184#   7478   10898
B800   = 000001         1102#
B801   = 000002         1101#
B802   = 000004         1100#
B803   = 000010         1099#
B804   = 000020         1098#
B805   = 000040         1097#
B806   = 000100         1096#
B807   = 000200         1095#
B808   = 000400         1094#
B809   = 001000         1093#
BIT0   = 000001          891#  10185
BIT00  = 000001          881#    891     917     966     985    1004    1042    1061    1102    1188    1204
BIT01  = 000002          880#    890     916     965    1003    1041    1060    1101    1187    1203
BIT02  = 000004          879#    889     915     964    1002    1040    1059    1100    1186    1202
BIT03  = 000010          878#    888     914     963    1001    1039    1058    1099    1114    1184    1201
BIT04  = 000020          877#    887     913     962    1000    1057    1098    1183
BIT05  = 000040          876#    886     912     999    1038    1056    1097    1182
BIT06  = 000100          875#    885     984     998    1020    1037    1055    1096    1167    1181    1200
BIT07  = 000200          874#    884     983     997    1019    1036    1054    1078    1095    1113    1166    1180
BIT08  = 000400          873#    883     961     982     996    1018    1035    1053    1094    1165    1178   13163
BIT09  = 001000          872#    882     960     981     995    1017    1034    1052    1093    1164    1177   13181   13273
BIT1   = 000002          890#   9653    9811   10915
BIT10  = 002000          871#    959     980     994    1016    1033    1051    1077    1092    1112    1163    1176    1199
                        13250
BIT11  = 004000          870#    911     979     993    1032    1050    1068    1076    1091    1111    1175    1198    9281
                        13188
BIT12  = 010000          869#    978     992    1031    1049    1075    1090    1110    1174    1197    9231
BIT13  = 020000          868#    977     991    1030    1048    1067    1089    1109    1161    1173    1196    6255    6609
                        13257
BIT14  = 040000          867#    976     990    1029    1047    1066    1088    1108    1119    1160    1172    1195    9196
                        13149
BIT15  = 100000          866#    975     989    1028    1046    1065    1087    1107    1118    1159    1171    1194    9183
                         9896    9950
BIT2   = 000004          889#   10185
BIT3   = 000010          888#   9309   12357   12444
BIT4   = 000020          887#   9259   12463
BIT5   = 000040          886#
BIT6   = 000100          885#   9208
BIT7   = 000200          884#   3265   10191
BIT8   = 000400          883#
BIT9   = 001000          882#
BOTADR  037216          9047*   9065*   9068    9083    9128#
BOTFLG  037220          9033*   9075*   9078    9081*   9129#
BPTVEC= 000014            898#
BSE    = 100000         1107#   6448    6459    6579    6590    6780    6783    6871    6874    9559   10802
BUFFER  106520          9378*   9627   13730#
BUFONE  106520          3597    3680    3817    3854    3937    4102    4186    4394    4477    4614    4651    4734    4899
                        4983    5191    5255    5444    5558    5720    5760    5824    6013    6078    6270    6288    6290*
                        6292*   6360    6644    6662    6664*   6666*   6736    6896    6929    6982    7168    7393    7481
                        7621    7658    7742    7879    7922    8006    8170    8254    8390    8427    8511    8647    8684
                        8767    8904   13730#
BUFTWO  107524          3760    3818    4254*   4320    4335    4337    4358*   4557    4615    5051*   5117    5132    5134
                        5155*   5666    5721    6492    6827    6897    7087    7564    7622    7821    7880    8334    8391
```

```
                        8590    8648    8847    8905   13730#
CC     = 004000         1091#
CH     = 002000         1092#
CHRCNT  037221          9034*   9052*   9058*   9059    9062*   9066    9071*   9082*   9130#
CKSWR  = 104410        13147   13246   13272   13623#
CLEAN  = ****** U       6253    6607   13730
CLKADR  001500          1447#  10147*  10149   10153*  10155   10180
CLKVCT  001502          1448#  10148*  10154*
CLR    = 000040         1182#  11341
CLRSTS  052256          9218   11377#
CLSPRN  067014          3317    3333    3354   13730#
CMNSTA  006610          3271    3403#
CMPBUF  042344          3816    4613    5719    6895    7620    7878    8389    8646    8903    9876#
CMPERR  062402         12756#
CNSL00  067154          3281   13730#
CNSL01  067213          3314   13730#
CNSL02  067240          3323   13730#
CNSL03  067321          3328   13730#
CNSL04  067351          3339   13730#
CNSL05  067425          3349   13730#
CNSL06  067461          3360   13730#
CNSL07  067506          3373   13730#
CNTCLR  052140          3502    3546    9198   11333#
CONT   = 000100         1055#
CR     = 000015          806#   3386    9050   12835   13122   13132   13730
CRLF   = 000200          807#   3254   13093   13132
CYLMSK= 001777          1082#   9814
DBCK   = 100000         1028#
DBEN   = 040000         1029#
DBL    = 002000         1199#
DCK    = 100000          989#   1006    9552    9884   10840   10860   12450   12453
DDISP  = 177570          813#   1299    3218
DEBL   = 020000         1030#
DEVSEL  050466          9185   11043#
DISPLA  001156          1299#   3218*   3226*  13203*  13249*
DISPRE  000174          1226#   3226
DLT    = 100000         1171#  10684   12529   12532
DMD    = 000001         1042#   1061#   6961    7067
DPE    = 000010         1114#   9553    9559   10346   10626   11122   11661   12082   12503   12506
DPEHI  = 040000         1195#
DPELO  = 020000         1196#
DPR    = 000400          982#  11283   11480   11481   11947
DRQ    = 004000         1068#
DRVCLR= 000010           926#   9406   11935
DRVSTS  055474          9424   11926#
DRY    = 000200          983#  10313   10409   10412   10413   11283   11480   11481   11947
DSWR   = 177570          812#   1298    3217
DTASTS  056276          3648    3721    3800    3905    3978    4056    4153    4226    4309    4445    4518    4597    4702
                        4775    4853    4950    5023    5106    5242    5320    5398    5495    5595    5703    5811    5889
                        5967    6064    6142    6220    6328    6415    6434    6451    6546    6565    6582    6702    6788
                        6879    7023    7128    7444    7522    7604    7709    7782    7862    7973    8046    8124    8221
                        8294    8373    8478    8551    8630    8735    8808    8887    9527   12057#
DTE    = 010000          992#   1006   12316   12334   12337
DTO    = 010000         1031#
DULPRT= 024024          1071#   3561    3564
DVA    = 004000          911#   3521   10036   10109   10274   10277   11068   11386   11934   11935
```

```
DVC    = 000200          1113#  11215  11470  11722  11760  11763  11896  12269  12273  12276  12321  12695  12700
                        12703
EARLY    070272           8183  13730#
EBL    = 020000          1048#
ECH    = 000100           998#   1006   9552   9886  10823  10838  10856  10862  12460  13730
ECI    = 004000          1076#   9888  10858  12457
ECRC   = 001000          1052#
EDT1     074622          1482   1489   1496   1503   1510   1517   1531   1538   1545   1552   1559   1566   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   1777   1784   1791   1798   1805   1813   1821   1829   1843   1850   1857
                         1864   1871   1878   1885   1893   1900   1907   1914   1921   1928   1935   1942   1949
                         1956   1963   1970   1977   2019   2026   2033   2040   2047   2054   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   2293   2300   2307   2314   2328   2335   2342   2349   2356   2363
                         2370   2377   2384   2391   2398   2405   2412   2419   2426   2433   2440   2447   2454
                         2475   2482   2489   2496   2503   2643   2650   2657   2678   2685   2692   2706   2713
                         2720   2727   2734   2741   2748   2755   2763   2770   2778   2785   2792   2800   2808
                         2816   2825   2833   2841   2848   2855   2862   2869   2876   2883   2890   2897   2904
                         2911   2918   2925   2932   2939   2946   2953   2960   2967   2974   2981   2988   2995
                         3002   3009   3016   3023   3030   3037   3072   3079   3093   3100   3107   3114   3121
                         3142  13730#
EDT110   074640          1984  13730#
EDT111   074642          1991   1998  13730#
EDT114   074644          2012  13730#
EDT2     074632          2461   2468   2664   2671  13730#
EDT223   074646          2510   2699  13730#
EDT336   074656          2286   3044   3058   3065  13730#
EDT337   074666          3051  13730#
EDT344   074676          3086  13730#
EDT353   074710          3135  13730#
ED1      104274         13730#
ED110    104302         13730#
ED111    104306         13730#
ED114    104314         13730#
ED223    104324         13730#
ED336    104334         13730#
ED337    104346         13730#
ED353    104360         13730#
EECC   = 000020          1057#
EFT1     074712          1483   1490   1497   1504   1511   1518   1532   1539   1546   1553   1560   1567   1574
                         1581   1588   1595   1602   1609   1616   1623   1630   1637   1644   1651   1658   1665
                         1672   1679   1686   1693   1700   1707   1714   1721   1728   1735   1742   1749   1756
                         1763   1770   1778   1785   1792   1799   1806   1814   1822   1830   1844   1851   1858
                         1865   1872   1879   1886   1894   1901   1908   1915   1922   1929   1936   1943   1950
                         1957   1964   1971   1978   2020   2027   2034   2041   2048   2055   2062   2069   2076
                         2083   2090   2097   2104   2111   2118   2125   2132   2139   2146   2153   2160   2167
                         2174   2181   2188   2195   2202   2209   2216   2223   2230   2237   2244   2251   2258
                         2265   2272   2279   2294   2301   2308   2315   2329   2336   2343   2350   2357   2364
                         2371   2378   2385   2392   2399   2406   2413   2420   2427   2434   2441   2448   2455
                         2476   2483   2490   2497   2504   2644   2651   2658   2679   2686   2693   2707   2714
                         2721   2728   2735   2742   2749   2756   2764   2771   2779   2786   2793   2801   2809
                         2817   2826   2834   2842   2849   2856   2863   2870   2877   2884   2891   2898   2905
                         2912   2919   2926   2933   2940   2947   2954   2961   2968   2975   2982   2989   2996
                         3003   3010   3017   3024   3031   3038   3073   3080   3094   3101   3108   3115   3122
```

```
                    3143  13730#
                    1985  13730#
EFT110  074730
EFT111  074732      1992  1999  13730#
EFT114  074734      2013  13730#
EFT2    074722      2462  2469  2665  2672  13730#
EFT223  074736      2511  2700  13730#
EFT336  074746      2287  3045  3059  3066  13730#
EFT337  074756      3052  13730#
EFT344  074766      3087  13730#
EFT353  074776      3136  13730#
EF110   104452     13730#
EF111   104453     13730#
EF114   104455     13730#
EF336   104460     13730#
EF337   104464     13730#
EHT1    074476      1481  1488  1495  1502  1509  1516  1530  1537  1544  1551  1558  1565  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  1776  1783  1790  1797  1804  1812  1820  1828  1842  1849  1856
                    1863  1870  1877  1884  1892  1899  1906  1913  1920  1927  1934  1941  1948
                    1955  1962  1969  1976  2018  2025  2032  2039  2046  2053  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  2292  2299  2306  2313  2327  2334  2341  2348  2355  2362
                    2369  2376  2383  2390  2397  2404  2411  2418  2425  2432  2439  2446  2453
                    2474  2481  2488  2495  2502  2642  2649  2656  2677  2684  2691  2705  2712
                    2719  2726  2733  2740  2747  2754  2762  2769  2777  2784  2791  2799  2807
                    2815  2824  2832  2840  2847  2854  2861  2868  2875  2882  2889  2896  2903
                    2910  2917  2924  2931  2938  2945  2952  2959  2966  2973  2980  2987  2994
                    3001  3008  3015  3022  3029  3036  3071  3078  3092  3099  3106  3113  3120
                    3141  13730#
EHT110  074520      1983  13730#
EHT111  074524      1990  1997  13730#
EHT114  074530      2011  13730#
EHT2    074510      2460  2467  2663  2670  13730#
EHT223  074534      2509  13730#
EHT256  074546      2698  13730#
EHT336  074560      2285  3043  3057  3064  13730#
EHT337  074572      3050  13730#
EHT344  074604      3085  13730#
EHT353  074616      3134  13730#
EH1     103227     13730#
EH110   103246     13730#
EH111   103255     13730#
EH114   103274     13730#
EH223   103323     13730#
EH256   103351     13730#
EH336   103426     13730#
EH337   103465     13730#
EH344   103624     13730#
EH353   103764     13730#
EMS1    075000     13730#
EMS10   075334     13730#
EMS100  077554     13730#
EMS101  077602     13730#
EMS102  077630     13730#
```

```
EMS103  077657          13730#
EMS104  077666          13730#
EMS105  077715          13730#
EMS106  077744          13730#
EMS11   075376          13730#
EMS110  077763          13730#
EMS111  100012          13730#
EMS112  100023          13730#
EMS113  100107          13730#
EMS114  100137          13730#
EMS115  100163          13730#
EMS116  100170          13730#
EMS117  100220          13730#
EMS12   075407          13730#
EMS120  100242          13730#
EMS121  100257          13730#
EMS122  100322          13730#
EMS123  100357          13730#
EMS124  100422          13730#
EMS125  100454          13730#
EMS126  100517          13730#
EMS127  100562          13730#
EMS13   075450          13730#
EMS130  100621          13730#
EMS131  100657          13730#
EMS132  100717          13730#
EMS133  100743          13730#
EMS134  100766          13730#
EMS135  101030          13730#
EMS136  101072          13730#
EMS137  101117          13730#
EMS14   075461          13730#
EMS140  101154          13730#
EMS141  101175          13730#
EMS142  101223          13730#
EMS143  101240          13730#
EMS144  101266          13730#
EMS145  101276          13730#
EMS146  101325          13730#
EMS147  101344          13730#
EMS15   075472          13730#
EMS150  101361          13730#
EMS151  101400          13730#
EMS152  101424          13730#
EMS153  101450          13730#
EMS154  101472          13730#
EMS155  101506          13730#
EMS156  101526          13730#
EMS157  101544          13730#
EMS16   075514          13730#
EMS160  101561          13730#
EMS161  101576          13730#
EMS162  101626          13730#
EMS163  101661          13730#
EMS164  101705          13730#
EMS165  101766          13730#
```

```
EMS166  102002          13730#
EMS167  102027          13730#
EMS17   075544          13730#
EMS170  102054          13730#
EMS171  102103          13730#
EMS172  102136          13730#
EMS173  102156          13730#
EMS174  102173          13730#
EMS175  102201          13730#
EMS176  102227          13730#
EMS177  102256          13730#
EMS2    075047          13730#
EMS20   075604          13730#
EMS200  102300          13730#
EMS201  102315          13730#
EMS202  102363          13730#
EMS203  102413          13730#
EMS204  102426          13730#
EMS205  102454          13730#
EMS206  102466          13730#
EMS207  102513          13730#
EMS21   075627          13730#
EMS210  102540          13730#
EMS211  102551          13730#
EMS212  102567          13730#
EMS213  102612          13730#
EMS214  102635          13730#
EMS215  102653          13730#
EMS216  102674          13730#
EMS217  102724          13730#
EMS22   075652          13730#
EMS220  102757          13730#
EMS221  102777          13730#
EMS222  103047          13730#
EMS223  103100          13730#
EMS224  103155          13730#
EMS23   075666          13730#
EMS24   075714          13730#
EMS25   075743          13730#
EMS26   075760          13730#
EMS27   076001          13730#
EMS3    075064          13730#
EMS30   076012          13730#
EMS31   076022          13730#
EMS32   076051          13730#
EMS33   076100          13730#
EMS34   076126          13730#
EMS35   076177          13730#
EMS36   076226          13730#
EMS37   076255          13730#
EMS4    075127          13730#
EMS40   076303          13730#
EMS41   076332          13730#
EMS42   076360          13730#
EMS43   076407          13730#
EMS44   076436          13730#
```

```
EMS45   076511          13730#
EMS46   076554          13730#
EMS47   076603          13730#
EMS5    075172          13730#
EMS50   076632          13730#
EMS51   076661          13730#
EMS52   076707          13730#
EMS53   076721          13730#
EMS54   076733          13730#
EMS55   076755          13730#
EMS56   077004          13730#
EMS57   077061          13730#
EMS6    075222          13730#
EMS60   077107          13730#
EMS61   077122          13730#
EMS62   077151          13730#
EMS63   077167          13730#
EMS64   077215          13730#
EMS65   077233          13730#
EMS66   077301          13730#
EMS67   077351          13730#
EMS7    075267          13730#
EMS70   077376          13730#
EMS71   077410          13730#
EMS72   077423          13730#
EMS73   077433          13730#
EMS74   077450          13730#
EMS75   077467          13730#
EMS76   077475          13730#
EMS77   077526          13730#
EMTVEC= 000030            901#    3201*    3202*
EMT1    070404           1480   13730#
EMT10   070454           1529   13730#
EMT100  071452           1926   13730#
EMT101  071470           1933   13730#
EMT102  071506           1940   13730#
EMT103  071516           1947   13730#
EMT104  071530           1954   13730#
EMT105  071546           1961   13730#
EMT106  071556           1968   13730#
EMT107  071566           1975   13730#
EMT11   070460           1536   13730#
EMT110  071606           1982   13730#
EMT111  071620           1989   13730#
EMT112  071626           1996   13730#
EMT113  071634           2003   13730#
EMT114  071650           2010   13730#
EMT115  071656           2017   13730#
EMT116  071666           2024   13730#
EMT117  071676           2031   13730#
EMT12   070464           1543   13730#
EMT120  071706           2038   13730#
EMT121  071716           2045   13730#
EMT122  071726           2052   13730#
EMT123  071736           2059   13730#
EMT124  071746           2066   13730#
```

```
EMT125  071756            2073   13730#
EMT126  071766            2080   13730#
EMT127  072000            2087   13730#
EMT13   070472            1550   13730#
EMT130  072012            2094   13730#
EMT131  072024            2101   13730#
EMT132  072036            2108   13730#
EMT133  072050            2115   13730#
EMT134  072062            2122   13730#
EMT135  072074            2129   13730#
EMT136  072106            2136   13730#
EMT137  072120            2143   13730#
EMT14   070504            1557   13730#
EMT140  072130            2150   13730#
EMT141  072140            2157   13730#
EMT142  072150            2164   13730#
EMT143  072160            2171   13730#
EMT144  072170            2178   13730#
EMT145  072200            2185   13730#
EMT146  072210            2192   13730#
EMT147  072220            2199   13730#
EMT15   070512            1564   13730#
EMT150  072230            2206   13730#
EMT151  072240            2213   13730#
EMT152  072252            2220   13730#
EMT153  072264            2227   13730#
EMT154  072302            2234   13730#
EMT155  072320            2241   13730#
EMT156  072332            2248   13730#
EMT157  072344            2255   13730#
EMT16   070520            1571   13730#
EMT160  072356            2262   13730#
EMT161  072366            2269   13730#
EMT162  072400            2276   13730#
EMT163  072412          13730#
EMT164  072422            2291   13730#
EMT165  072430            2298   13730#
EMT166  072436            2305   13730#
EMT167  072444            2312   13730#
EMT17   070530            1578   13730#
EMT170  072452          13730#
EMT171  072462            2326   13730#
EMT172  072472            2333   13730#
EMT173  072502            2340   13730#
EMT174  072512            2347   13730#
EMT175  072524            2354   13730#
EMT176  072532            2361   13730#
EMT177  072540            2368   13730#
EMT2    070410            1487   13730#
EMT20   070540            1585   13730#
EMT200  072552            2375   13730#
EMT201  072564            2382   13730#
EMT202  072576            2389   13730#
EMT203  072610            2396   13730#
EMT204  072622            2403   13730#
EMT205  072640            2410   13730#
```

```
EMT206  072650            2417   13730#
EMT207  072660            2424   13730#
EMT21   070550            1592   13730#
EMT210  072672            2431   13730#
EMT211  072700            2438   13730#
EMT212  072714            2445   13730#
EMT213  072730            2452   13730#
EMT214  072740            2459   13730#
EMT215  072760            2466   13730#
EMT216  072772            2473   13730#
EMT217  073002            2480   13730#
EMT22   070560            1599   13730#
EMT220  073012            2487   13730#
EMT221  073022            2494   13730#
EMT222  073032            2501   13730#
EMT223  073042            2508   13730#
EMT224  073052          13730#
EMT225  073054          13730#
EMT226  073056          13730#
EMT227  073060          13730#
EMT23   070570            1606   13730#
EMT230  073062          13730#
EMT231  073064          13730#
EMT232  073066          13730#
EMT233  073070          13730#
EMT234  073072          13730#
EMT235  073074          13730#
EMT236  073076          13730#
EMT237  073100          13730#
EMT24   070600            1613   13730#
EMT240  073102          13730#
EMT241  073104          13730#
EMT242  073106          13730#
EMT243  073110          13730#
EMT244  073112          13730#
EMT245  073114          13730#
EMT246  073116            2641   13730#
EMT247  073126            2648   13730#
EMT25   070610            1620   13730#
EMT250  073140            2655   13730#
EMT251  073154            2662   13730#
EMT252  073162            2669   13730#
EMT253  073170            2676   13730#
EMT254  073206            2683   13730#
EMT255  073216            2690   13730#
EMT256  073226            2697   13730#
EMT257  073236            2704   13730#
EMT26   070620            1627   13730#
EMT260  073250            2711   13730#
EMT261  073262            2718   13730#
EMT262  073302            2725   13730#
EMT263  073312            2732   13730#
EMT264  073322            2739   13730#
EMT265  073342            2746   13730#
EMT266  073362            2753   13730#
EMT267  073374            2761   13730#
```

```
EMT27   070630          1634   13730#
EMT270  073412          2768   13730#
EMT271  073426          2776   13730#
EMT272  073444          2783   13730#
EMT273  073462          2790   13730#
EMT274  073500          2798   13730#
EMT275  073516          2806   13730#
EMT276  073530          2814   13730#
EMT277  073544          2823   13730#
EMT3    070416          1494   13730#
EMT30   070640          1641   13730#
EMT300  073562          2831   13730#
EMT301  073602          2839   13730#
EMT302  073624          2846   13730#
EMT303  073636          2853   13730#
EMT304  073650          2860   13730#
EMT305  073662          2867   13730#
EMT306  073674          2874   13730#
EMT307  073704          2881   13730#
EMT31   070650          1648   13730#
EMT310  073724          2888   13730#
EMT311  073736          2895   13730#
EMT312  073750          2902   13730#
EMT313  073762          2909   13730#
EMT314  074002          2916   13730#
EMT315  074014          2923   13730#
EMT316  074026          2930   13730#
EMT317  074040          2937   13730#
EMT32   070660          1655   13730#
EMT320  074050          2944   13730#
EMT321  074060          2951   13730#
EMT322  074070          2958   13730#
EMT323  074076          2965   13730#
EMT324  074106          2972   13730#
EMT325  074120          2979   13730#
EMT326  074132          2986   13730#
EMT327  074150          2993   13730#
EMT33   070670          1662   13730#
EMT330  074162          3000   13730#
EMT331  074172          3007   13730#
EMT332  074204          3014   13730#
EMT333  074216          3021   13730#
EMT334  074234          3028   13730#
EMT335  074252          3035   13730#
EMT336  074272          2284     3042   13730#
EMT337  074302          3049   13730#
EMT34   070700          1669   13730#
EMT340  074312          3056   13730#
EMT341  074324          3063   13730#
EMT342  074332          3070   13730#
EMT343  074344          3077   13730#
EMT344  074356          3084   13730#
EMT345  074370          3091   13730#
EMT346  074400          3098   13730#
EMT347  074414          3105   13730#
EMT35   070710          1676   13730#
```

```
EMT350  074426          3112   13730#
EMT351  074444          3119   13730#
EMT352  074454          3126   13730#
EMT353  074460          3133   13730#
EMT354  074470          3140   13730#
EMT36   070720          1683   13730#
EMT37   070730          1690   13730#
EMT4    070424          1501   13730#
EMT40   070740          1697   13730#
EMT41   070746          1704   13730#
EMT42   070756          1711   13730#
EMT43   070770          1718   13730#
EMT44   071000          1725   13730#
EMT45   071010          1732   13730#
EMT46   071020          1739   13730#
EMT47   071030          1746   13730#
EMT5    070432          1508   13730#
EMT50   071036          1753   13730#
EMT51   071046          1760   13730#
EMT52   071060          1767   13730#
EMT53   071076          1775   13730#
EMT54   071114          1782   13730#
EMT55   071124          1789   13730#
EMT56   071136          1796   13730#
EMT57   071154          1803   13730#
EMT6    070440          1515   13730#
EMT60   071164          1811   13730#
EMT61   071202          1819   13730#
EMT62   071222          1827   13730#
EMT63   071236         13730#
EMT64   071240          1841   13730#
EMT65   071260          1848   13730#
EMT66   071300          1855   13730#
EMT67   071312          1862   13730#
EMT7    070446          1522   13730#
EMT70   071324          1869   13730#
EMT71   071334          1876   13730#
EMT72   071344          1883   13730#
EMT73   071362          1891   13730#
EMT74   071400          1898   13730#
EMT75   071410          1905   13730#
EMT76   071430          1912   13730#
EMT77   071440          1919   13730#
ENRGDT  070404         13730#
EQUALS  067015         13730#
ERR     = 040000        976#  10455   10457   10473   10507
ERRNMB  037214         9014*  9015*   9018    9030    9035    9127#
ERRTYP  036456         8993#  13259
ERRVEC= 000004          894#   3215    3216*   3227*   3462    3463    3464*   3465*   3476*   3477*   3481*   3482*   3508
                       3509   3510*   3511*   3515*   3516*   3527*   3528*   10027   10028   10032*  10033*  10064*  10065*
                      10100  10101   10105*  10106*  10131*  10132*  10143   10144   10145*  10146*  10152*  10160*  10161*
                      10175  10176   10177*  10178*  10202*  10203*  11050   11051   11054*  11055*  11088*  11089*  11336
                      11337  11338*  11339*  11350*  11351*  13154   13155*  13157*  13160*
ERTY00  037222         9000   9132#
ERTY01  037230         9008   9133#
ERTY02  037240         9017   9135#
```

ERTY03 037247
ERTY04 037255
ESRC = 004000
FER = 000020

FIND = 000001

9023 9137#
9119 9139#
1050#
1000# 1006 6431 6441 6562 6572 6775 6778 6866 6869 9552 10787 12385
12402 12405 12428 12431
3586# 3588# 3604# 3605 3626# 3627 3634# 3637# 3640# 3641 3648# 3649 3668#
3670# 3692# 3693 3700# 3703# 3706# 3707 3714# 3715 3721# 3722 3730# 3731
3749# 3751# 3771# 3772 3779# 3782# 3785# 3786 3793# 3794 3800# 3801 3809#
3810 3816# 3819 3843# 3845# 3861# 3862 3883# 3884 3891# 3894# 3897# 3898
3905# 3906 3925# 3927# 3949# 3950 3957# 3960# 3963# 3964 3971# 3972 3978#
3979 3987# 3988 4006# 4008# 4027# 4028 4035# 4038# 4041# 4042 4049# 4050
4056# 4057 4065# 4066 4091# 4093# 4109# 4110 4131# 4132 4139# 4142# 4145#
4146 4153# 4154 4173# 4175# 4197# 4198 4205# 4208# 4211# 4212 4219# 4220
4226# 4227 4235# 4236 4257# 4259# 4278# 4279 4286# 4289# 4292# 4293 4300#
4301 4309# 4310 4350# 4351 4383# 4385# 4401# 4402 4423# 4424 4431# 4434#
4437# 4438 4445# 4446 4465# 4467# 4489# 4490 4497# 4500# 4503# 4504 4511#
4512 4518# 4519 4527# 4528 4546# 4548# 4568# 4569 4576# 4579# 4582# 4583
4590# 4591 4597# 4598 4606# 4607 4613# 4616 4640# 4642# 4658# 4659 4680#
4681 4688# 4691# 4694# 4695 4702# 4703 4722# 4724# 4746# 4747 4754# 4757#
4760# 4761 4768# 4769 4775# 4776 4784# 4785 4803# 4805# 4824# 4825 4832#
4835# 4838# 4839 4846# 4847 4853# 4854 4862# 4863 4888# 4890# 4906# 4907
4928# 4929 4936# 4939# 4942# 4943 4950# 4951 4970# 4972# 4994# 4995 5002#
5005# 5008# 5009 5016# 5017 5023# 5024 5032# 5033 5054# 5056# 5075# 5076
5083# 5086# 5089# 5090 5097# 5098 5106# 5107 5147# 5148 5180# 5182# 5198#
5199 5220# 5221 5228# 5231# 5234# 5235 5242# 5243 5269# 5271# 5291# 5292
5299# 5302# 5305# 5306 5313# 5314 5320# 5321 5329# 5330 5348# 5350# 5369#
5370 5377# 5380# 5383# 5384 5391# 5392 5398# 5399 5407# 5408 5433# 5435#
5451# 5452 5473# 5474 5481# 5484# 5487# 5488 5495# 5496 5518# 5520# 5529#
5530 5537# 5538# 5539# 5540 5545# 5546 5569# 5570 5577# 5580# 5581 5588#
5589 5595# 5596 5604# 5605 5626# 5628# 5637# 5638 5645# 5646# 5647# 5648
5653# 5654 5676# 5677 5685# 5688# 5689 5696# 5697 5703# 5704 5712# 5713
5719# 5722 5749# 5751# 5767# 5768 5789# 5790 5797# 5800# 5803# 5804 5811#
5812 5838# 5840# 5860# 5861 5868# 5871# 5874# 5875 5882# 5883 5889# 5890
5898# 5899 5917# 5919# 5938# 5939 5946# 5949# 5952# 5953 5960# 5961 5967#
5968 5976# 5977 6002# 6004# 6020# 6021 6042# 6043 6050# 6053# 6056# 6057
6064# 6065 6091# 6093# 6113# 6114 6121# 6124# 6127# 6128 6135# 6136 6142#
6143 6151# 6152 6170# 6172# 6191# 6192 6199# 6202# 6205# 6206 6213# 6214
6220# 6221 6229# 6230 6259# 6261# 6277# 6278 6306# 6307 6314# 6317# 6320#
6321 6328# 6329 6348# 6350# 6372# 6373 6380# 6383# 6386# 6387 6394# 6395
6415# 6416 6434# 6435 6451# 6452 6463# 6464 6481# 6483# 6504# 6505 6512#
6515# 6518# 6519 6526# 6527 6546# 6547 6565# 6566 6582# 6583 6594# 6595
6633# 6635# 6651# 6652 6680# 6681 6688# 6691# 6694# 6695 6702# 6703 6722#
6724# 6746# 6747 6754# 6757# 6760# 6761 6768# 6769 6788# 6789 6797# 6798
6815# 6817# 6837# 6838 6845# 6848# 6851# 6852 6859# 6860 6879# 6880 6888#
6889 6895# 6898 6935# 6950# 6952# 6992# 6993 7000# 7003# 7006# 7007 7014#
7015 7023# 7024 7037# 7038 7056# 7058# 7097# 7098 7105# 7108# 7111# 7112
7119# 7120 7128# 7129 7142# 7143 7174# 7188# 7190# 7222# 7223# 7224 7229#
7230 7243# 7246# 7247 7254# 7255 7262# 7263 7281# 7283# 7313# 7315# 7316
7323# 7324 7338# 7341# 7342 7349# 7350 7356# 7357 7382# 7384# 7400# 7401
7422# 7423 7430# 7433# 7436# 7437 7444# 7445 7464# 7466# 7493# 7494 7501#
7504# 7507# 7508 7515# 7516 7522# 7523 7531# 7532 7550# 7552# 7575# 7576
7583# 7586# 7589# 7590 7597# 7598 7604# 7605 7613# 7614 7620# 7623 7647#
7649# 7665# 7666 7687# 7688 7695# 7698# 7701# 7702 7709# 7710 7729# 7731#
7753# 7754 7761# 7764# 7767# 7768 7775# 7776 7782# 7783 7791# 7792 7810#
7812# 7833# 7834 7841# 7844# 7847# 7848 7855# 7856 7862# 7863 7871# 7872
7878# 7881 7911# 7913# 7929# 7930 7951# 7952 7959# 7962# 7965# 7966 7973#

```
                    7974    7993#   7995#   8017#   8018    8025#   8028#   8031#   8032    8039#   8040    8046#   8047
                    8055#   8056    8074#   8076#   8095#   8096    8103#   8106#   8109#   8110    8117#   8118    8124#
                    8125    8133#   8134    8159#   8161#   8177#   8178    8199#   8200    8207#   8210#   8213#   8214
                    8221#   8222    8241#   8243#   8265#   8266    8273#   8276#   8279#   8280    8287#   8288    8294#
                    8295    8303#   8304    8322#   8324#   8344#   8345    8352#   8355#   8358#   8359    8366#   8367
                    8373#   8374    8382#   8383    8389#   8392    8416#   8418#   8434#   8435    8456#   8457    8464#
                    8467#   8470#   8471    8478#   8479    8498#   8500#   8522#   8523    8530#   8533#   8536#   8537
                    8544#   8545    8551#   8552    8560#   8561    8579#   8581#   8601#   8602    8609#   8612#   8615#
                    8616    8623#   8624    8630#   8631    8639#   8640    8646#   8649    8673#   8675#   8691#   8692
                    8713#   8714    8721#   8724#   8727#   8728    8735#   8736    8755#   8757#   8779#   8780    8787#
                    8790#   8793#   8794    8801#   8802    8808#   8809    8817#   8818    8836#   8838#   8858#   8859
                    8866#   8869#   8872#   8873    8880#   8881    8887#   8888    8896#   8897    8903#   8906
FMT16 = 010000      1075#   3599    3856    4104    4396    4653    4901    5193    5446    5762    6015    6272    6405
                    6536    6646    6734    6931    7170    7395    7660    7924    8172    8429    8686    9388    9817
                    9819    9890    11147   12223
FNCDTB  067646      10501   13730#
FNCMSK= 000077        918#  11934   12378
F0    = 000002        916#  10400
F1    = 000004        915#  10400
F2    = 000010        914#  10400   10974
F3    = 000020        913#  10400
F4    = 000040        912#  10400
GENBUF  042100      3612    3869    4117    4409    4666    4914    5206    5259    5459    5775    5828    6028    6081
                    6285    6659    7408    7673    7937    8185    8442    8699    9801#
GET     042776      3555    3640    3706    3785    3897    3963    4041    4145    4211    4292    4327    4437    4503
                    4582    4694    4760    4838    4942    5008    5089    5125    5234    5305    5383    5487    5539
                    5580    5647    5688    5803    5874    5952    6056    6127    6205    6320    6386    6518    6694
                    6760    6851    7006    7111    7229    7246    7323    7341    7436    7507    7589    7701    7767
                    7847    7965    8031    8109    8213    8279    8358    8470    8536    8615    8727    8793    8872
                    9211    9235    9262    9285    9311    9415    9448    9484    9518    10018#
GETBUF  001326      1382#   10030
GETINX  001504      1453#   3552*   3553*   4325*   4326*   5122*   5123*   9233*   9234*   9283*   9284*   9976    10031
GETSTS  042712      3634    3700    3779    3891    3957    4035    4139    4205    4286    4431    4497    4576    4688
                    4754    4832    4936    5002    5083    5228    5299    5377    5481    5537    5645    5797    5868
                    5946    6050    6121    6199    6314    6380    6512    6688    6754    6845    6935    7000    7105
                    7174    7222    7313    7430    7501    7583    7695    7761    7841    7959    8025    8103    8207
                    8273    8352    8464    8530    8609    8721    8787    8866    9210    9261    9304    9402    9972#
GNS   = ****** U    1225    3253    8952    8959    13614   13615   13616   13617   13618   13619   13621   13623   13624
                    13625   13626   13627   13628
GO    = 000001        917#  3600    3682    3761    3857    3939    4017    4105    4187    4268    4397    4479    4558
                    4654    4736    4814    4902    4984    5065    5194    5281    5359    5447    5528    5559    5636
                    5665    5763    5850    5928    6016    6103    6181    6273    6362    6494    6647    6735    6826
                    6981    7086    7171    7210    7302    7396    7482    7565    7661    7743    7823    7925    8007
                    8085    8173    8255    8333    8430    8512    8591    8687    8769    8848    9245    9294    9406
                    9439    9473    9507    10188   10299   10311   10316   10332   10422   10425   12379
GTSWR = 104407      3248    3437    13621#
HCE   = 000200        997#  1006    6412    6423    6543    6554    6775    6778    6866    6869    9552    10750   10755
                    10770   10772   10785   10800   12385   12415   12418   12428   12431   13730
HCI   = 002000      1077#   6734    10748   12381
HCRC  = 000400        996#  1006    9552    10757   12385   12389   12392   12428   12431
HELP    106520      3301    13730#
HELPQS  067025      3295    13730#
HT    = 000011        804#  9056    13091   13132
IAE   = 002000        994#  10654   10951   10966   10992   11138   11155   11654   11708   11711   12212   12227   13730
IDXMSK= 000077      1153#   10044   10047   10117   10120
IE    = 000100      1167#   1200#
```

```
ILF    = 000001          1004#  10526  10528  11548  11608  11612  11654  11675  11678  12168  12185  12188  13730
ILF02  = 000002           923#
ILF24  = 000024           933#
ILF26  = 000026           934#
ILF30  = 000030           936#
ILF32  = 000032           937#
ILF34  = 000034           938#
ILF36  = 000036           939#
ILF40  = 000040           940#
ILF42  = 000042           941#
ILF44  = 000044           942#
ILF46  = 000046           943#
ILF54  = 000054           946#
ILF56  = 000056           947#
ILF64  = 000064           950#
ILF66  = 000066           951#
ILF74  = 000074           954#
ILF76  = 000076           955#
ILR    = 000002          1003#  11548  11594  11598  11862  11866  11869  12168  12171  12174
ILRG50= 000050           1139#
ILRG52= 000052           1140#
ILRG54= 000054           1141#
ILRG56= 000056           1142#
ILRG60= 000060           1143#
ILRG62= 000062           1144#
ILRG64= 000064           1145#
ILRG66= 000066           1146#
ILRG70= 000070           1147#
ILRG72= 000072           1148#
ILRG74= 000074           1149#
ILRG76= 000076           1150#
IOTVEC= 000020            899#  3199*  3200*
IPCK0  = 000001          1204#
IPCK1  = 000002          1203#
IPCK2  = 000004          1202#
IPCK3  = 000010          1201#
IR     = 000100          1181#  11411
IVC    = 010000          1110#   7021   7033   7126   7136  10569  10570  11173  11296  11722  11729  11732  11801
                        12151  12155  12583  12664  13730
LBC    = 002000          1112#  12269  12287  12292
LBT    = 002000           980#  10729  10953  10955
LF     = 000012           805#   9054  12835  13126  13132  13730
LS     = 000004          1059#  11435  11987
LSC    = 004000          1111#  12269  12303  12306
LST    = 000002          1060#  11435  11987
MCLK   = 004000          1032#
MCPE   = 020000          1161#  10299  10316  10329  10332  12066  12069
MDF    = 000100          1037#
MDPE   = 000400          1178#  10825  12475  12478
MEDENB  001472           1441#   3448*  9244*  9367   9623*
MFGFIL  104500           9389    9592   9605*  9612*  9673  13730#
MI     = 000004          1040#
MIXED   067756           8440    8697  13730#
MOC    = 000400          1035#
MOH    = 020000          1067#
MOL    = 010000           978#  9544  11234  11244  11245  11250  11255  11283  11534  11537  11698  11775  11776
```

```
                    11783  11788  11947  12128  12289  12542  12543  12565  12570  12641  12642  12646  12651
MRD    = 002000      1033#
MS     = 000040      1038#
MSC    = 000002      1041#
MSE    = 100000      1118#   6407   6538   9592   9815
MSER   = 000200      1036#
MUR    = 001000      1034#
MWD    = 000010      1058#  11436  11988
MWP    = 000010      1039#
MXF    = 001000      1177#  10714  12089  12103  12106
NDTMSK= 115760       1006#
NED    = 010000      1174#   3517  10281  10285  11063
NEM    = 004000      1175#  10699
NOP    = 000000       922#
NSA    = 100000      1065#
OCC    = 100000      1046#
OFD    = 000200      1078#
OFFSET= 000014        928#
OM     = 000001       985#  11775  11831  11834  11946  12490  12493
ONES     070016      4407   4664   4912   5257   5457   5773   6079   6657   7406   7935  13730#
OPE    = 020000      1109#
OPI    = 020000       991#  10553  11228  11252  11548  11566  11570  11654  11691  11694  11785  12123  12126
                    12567  12648  13730
OR     = 000200      1180#
PACACK= 000022        932#   9439
PAKACK= 000022        931#    932   9245
PAR    = 000010      1001#   9555  10344  10348  10360  11120  11654  11659  11664  12080  12085
PAT    = 000020      1183#
PDA    = 000400      1053#
PGE    = 002000      1176#
PGM    = 001000       981#  11946
PHA    = 000200      1054#
PIP    = 020000       977#   7234   7237   7328   7331   9292   9470  11244  11265  11270  11775  11846  11851
                    11946  12542  12548  12553  12641  12678  12683
PIRQ   = 177772       811#
PIRQVE= 000240        905#
PLFS   = 002000      1051#
PRIERR   043772      3714   3793   3971   4049   4219   4300   4511   4590   4768   4846   5016   5097   5313
                     5391   5588   5696   5882   5960   6135   6213   6394   6526   6768   6859   7014   7119
                     7254   7349   7515   7597   7775   7855   8039   8117   8287   8366   8544   8623   8801
                     8880  10247#
PROMPT   067017      3374   3382  13730#
PR0    = 000000       828#
PR1    = 000040       829#
PR2    = 000100       830#
PR3    = 000140       831#  12814
PR4    = 000200       832#
PR5    = 000240       833#
PR6    = 000300       834#   3186   3443   3465   3511  10033  10106  10146  10178  11055  11339  12820
PR7    = 000340       835#
PS     = 177776       808#    809
PSEL   = 002000      1163#
PSW    = 177776       809#
PUT      043246      3626   3692   3771   3883   3949   4027   4131   4197   4278   4423   4489   4568   4680
                     4746   4824   4928   4994   5075   5220   5291   5369   5473   5529   5569   5637   5676
                     5789   5860   5938   6042   6113   6191   6306   6372   6504   6680   6746   6837   6962
```

```
                    6971    6992    7068    7077    7097    7200    7223    7292    7315    7422    7493    7575    7687
                    7753    7833    7951    8017    8095    8199    8265    8344    8456    8522    8601    8713    8779
                    8858    9248    9297    9407    9440    9474    9508   10094#
PUTBUF  001376       1409#   9378    9627*  10103
PUTINX  001533       1458#   3615    3683    3762    3872    3940    4018    4120    4188    4269    4412    4480    4559
                    4669    4737    4815    4917    4985    5066    5209    5282    5360    5462    5560    5667    5778
                    5851    5929    6031    6104    6182    6295    6363    6495    6669    6737    6828    6959*   6960*
                    6968*   6969*   6983    7065*   7066*   7074*   7075*   7088    7197*   7198*   7211    7289*   7290*
                    7303    7411    7483    7566    7676    7744    7824    7940    8008    8086    8188    8256    8335
                    8445    8513    8592    8702    8770    8849    9246*   9247*   9295*   9296*   9391   10104   10351
PWRVEC= 000024        900#   3205*   3206*  13633*  13634*  13643*  13649*  13661*  13662*
QSTMRK  067023       3290    3392   13730#
RCLSTS  053732       9318    9493   11645#
RD      = 000070      952#   3761    4558    5665    6494    6826    7086    7302    7565    7823    8333    8591    8848
RDCHR   = 104411     3282    3296    3306    3375    3383   13486   13624#
RDLIN   = 104412    13556   13625#
RDOCT   = 104413     3318    3334    3355   13626#
RDY     = 000200     1166#  10188   10189   10295   10298   11386
READY   006742       3287    3441#   8927    8975
RECAL   = 000006      925#   9294    9473
RESREG= 104415       9123   13628#
RESVEC= 000010        895#
REX     = 010000     1049#
RG      = 040000     1047#
RGDTPT  067756      13730#
RH      = 000072      953#   9507
RIP     = 000020      930#
RLEASE= 000012        927#
RMAS    = 000016     1127#
RMASI   001344       1392#  11967   13730
RMASO   001414       1419#
RMBA    = 000004     1209#   3470*   3471    3618    3687    3766    3875    3944    4022    4123    4192    4273    4415
                    4484    4563    4672    4741    4819    4920    4989    5070    5212    5286    5364    5465    5564
                    5671    5781    5855    5933    6034    6108    6186    6298    6367    6499    6672    6741    6832
                    6987    7092    7215    7307    7414    7488    7570    7679    7748    7828    7943    8012    8090
                    8191    8260    8339    8448    8517    8596    8705    8774    8853    9396
RMBAE = 000050       1212#
RMBAI   001332       1387#   4338    5135   10901   10903   11397   13730
RMBAO   001402       1414#   3597*   3680*   3760*   3854*   3937*   4102*   4186*   4394*   4477*   4557*   4651*   4734*
                    4899*   4983*   5191*   5255*   5444*   5558*   5666*   5760*   5824*   6013*   6078*   6270*   6360*
                    6492*   6644*   6736*   6827*   6929*   6982*   7087*   7168*   7393*   7481*   7564*   7658*   7742*
                    7821*   7922*   8006*   8170*   8254*   8334*   8427*   8511*   8590*   8684*   8767*   8847*   9389*
                    9807    9894   10897   10900
RMCC    = 000036     1134#
RMCCI   001364       1400#
RMCCO   001434       1427#
RMCS1   = 000000     1123#   1207#   3467*   3513    3621    3689    3768    3878    3946    4024    4126    4194    4275
                    4418    4486    4565    4675    4743    4821    4923    4991    5072    5215    5288    5366    5468
                    5566    5673    5784    5857    5935    6037    6110    6188    6301    6369    6501    6675    6743
                    6834    6989    7094    7217    7309    7417    7490    7572    7682    7750    7830    7946    8014
                    8092    8194    8262    8341    8451    8519    8598    8708    8776    8855    9246    9295    9397
                   10035   10108   10187   11061
RMCS1I  001326       1385#  10274   10276   10278   10295   10297   10300   10311   10315   10317   10329   10331   10333
                   10422   10424   10435   10478   10884   11384   11933   12066   12068   12070   12089   13730
RMCS10  001376       1412#   3600*   3682*   3761*   3857*   3939*   4017*   4105*   4187*   4268*   4397*   4479*   4558*
                    4654*   4736*   4814*   4902*   4984*   5065*   5194*   5256*   5281*   5359*   5447*   5528*   5559*
```

```
                    5636*  5665*  5763*  5827*  5850*  5928*  6016*  6076*  6103*  6181*  6273*  6362*  6494*
                    6647*  6735*  6826*  6932*  6981*  7086*  7171*  7210*  7302*  7396*  7482*  7565*  7661*
                    7743*  7823*  7925*  8007*  8085*  8173*  8255*  8333*  8430*  8512*  8591*  8687*  8769*
                    8848*  9245*  9294*  9406*  9439*  9473*  9507*  9653   9811   10399  10915  12357  12377
RMCS2 = 000010       1210#  3472*  3473   3514   7484   10034  10107  11060*  11062  11341*  11343*
RMCS2I  001336       1389#  4307   4316   4318   5104   5113   5115   10256  10281  10283  10284  10470  10541
                    10642  10685  10700  10715  10826  10898  11406  12103  12105  12107  12475  12477  12479
                    12529  12531  12533  13730
RMCS20  001406       1416#  7479*  7563*
RMCS3 = 000052       1213#
RMDA  = 000006       1124#  3617   3684   3763   3874   3941   4019   4122   4189   4270   4414   4481   4560
                     4671   4738   4816   4919   4986   5067   5211   5283   5361   5464   5561   5668   5780
                     5852   5930   6033   6105   6183   6297   6364   6496   6671   6738   6829   6984   7089
                     7212   7304   7413   7485   7567   7678   7745   7825   7942   8009   8087   8190   8257
                     8336   8447   8514   8593   8704   8771   8850   9392
RMDAI   001334       1388#  10997  13730
RMDAO   001404       1415#  3596*  3853*  4101*  4393*  4650*  4898*  5190*  5443*  5759*  6012*  6269*  6643*
                     6928*  7167*  7392*  7657*  7921*  8169*  8426*  8683*  8911*  8912   9385*  9566*  9567
                     9569   9587   9595*  9647   9774*  9810   10923  10924  11143  11145  11149  12217  12219
                    12221
RMDB  = 000022       1211#  3474*  3475   4325   5122   9983
RMDBI   001350       1394#  4334   5124*  5131
RMDBO   001420       1421#
RMDC  = 000034       1133#  3616   3685   3765   3873   3942   4021   4121   4190   4272   4413   4482   4562
                     4670   4739   4818   4918   4987   5069   5210   5284   5363   5463   5562   5670   5779
                     5853   5932   6032   6106   6185   6296   6365   6498   6670   6739   6831   6985   7091
                     7213   7306   7412   7486   7569   7677   7746   7827   7941   8010   8089   8189   8258
                     8338   8446   8515   8595   8703   8772   8852   9393
RMDCI   001362       1399#  11009  13730
RMDCO   001432       1426#  3595*  3852*  4100*  4392*  4649*  4897*  5189*  5442*  5506   5510*  5526   5527*
                     5551*  5615   5618*  5634   5635*  5659*  5758*  6011*  6268*  6642*  6927*  7166*  7391*
                     7656*  7886*  7887   7920*  8168*  8425*  8682*  9386*  9646   9773*  9809   10925  11140
                    12214
RMDS  = 000012       1125#  9233   9283
RMDSI   001340       1390#  7234   7236   7238   7328   7330   7332   9242   9292   9437   9470   9544   10313
                    10409  10411  10456  10473  10507  10511  10567  10594  10729  10954  11179  11234  11243
                    11250  11254  11256  11265  11269  11271  11280  11282  11284  11294  11298  11300  11479
                    11520  11522  11524  11534  11536  11538  11698  11736  11774  11783  11787  11789  11799
                    11803  11804  11815  11817  11819  11831  11833  11835  11846  11850  11852  11945  12128
                    12130  12159  12289  12355  12490  12492  12494  12541  12548  12552  12554  12565  12569
                    12571  12581  12585  12587  12640  12646  12650  12652  12662  12666  12667*  12668  12678
                    12682  12684  13730
RMDSO   001410       1417#
RMDT  = 000026       1130#  3552
RMDTI   001354       1396#  3554*  3559   3561   13730
RMDTO   001424       1423#
RMEC1 = 000044       1137#
RMEC1I  001372       1403#  9895   13730
RMEC1O  001442       1430#
RMEC2 = 000046       1138#  9978
RMEC2I  001374       1404#  9916   9977   11447  12011  13730
RMEC2O  001444       1431#
RMER1 = 000014       1126#  7197   7289   10352
RMER1I  001342       1391#  6412   6422   6431   6442   6543   6553   6562   6573   6775   6777   6866   6868
                     9551   9884   9886   10344  10359  10361  10451  10527  10554  10597  10612  10655  10732
                    10758  10773  10788  10841  10860  10863  10951  10966  10981  10992  11120  11127  11154
```

```
                    11228   11231   11252   11424   11548   11552   11554   11555   11566   11568   11569   11580   11582
                    11583   11594   11596   11597   11608   11610   11611   11654   11659   11663   11665   11675   11677
                    11679   11691   11693   11695   11708   11710   11712   11785   11862   11866   11868   11870   11880
                    11882   11884   11894   11898   11900   11958   12080   12084   12086   12123   12125   12127   12168
                    12171   12173   12175   12185   12187   12189   12199   12201   12203   12226   12316   12319   12323
                    12325   12334   12336   12338   12348   12350   12351   12385   12389   12391   12393   12402   12404
                    12406   12415   12417   12419   12428   12430   12432   12450   12452   12454   12460   12516   12518
                    12520   12567   12648   12759   13730
RMER10  001412       1418#   7199*   7291*  10348
RMER2 = 000042       1136#
RMER2I  001370       1402#   6448    6458    6579    6589    6780    6782    6871    6873    7021    7031    7032    7126
                     7135    7137    9553    9558   10346   10453   10571   10627   10803   11122   11173   11176   11187
                    11215   11218   11267   11296   11469   11661   11722   11729   11731   11733   11746   11748   11750
                    11760   11762   11764   11801   11848   11896   12020   12082   12151   12154   12156   12240   12244
                    12269   12273   12275   12277   12287   12291   12293   12303   12305   12307   12321   12503   12505
                    12507   12550   12583   12664   12680   12694   12700   12702   12704   12714   12716   12718   12762
                    13730
RMER20  001440       1429#
RMLA  = 000020       1128#
RMLAI   001346       1393#  13730
RMLAO   001416       1420#
RMMR1 = 000024       1129#   6959    6968    7065    7074
RMMR1I  001352       1395#  11434   11986   13730
RMMR10  001422       1422#   6961*   6970*   7067*   7076*
RMMR2 = 000040       1135#
RMMR2I  001366       1401#  11456   11998   13730
RMMR20  001436       1428#
RMOF  = 000032       1132#   3620    3686    3764    3877    3943    4020    4125    4191    4271    4417    4483    4561
                     4674    4740    4817    4922    4988    5068    5214    5285    5362    5467    5563    5669    5783
                     5854    5931    6036    6107    6184    6300    6366    6497    6674    6740    6830    6986    7090
                     7214    7305    7416    7487    7568    7681    7747    7826    7945    8011    8088    8193    8259
                     8337    8450    8516    8594    8707    8773    8851    9395
RMOFI   001360       1398#   9888    9890   10748   10858   12381   12457   13730
RMOFO   001430       1425#   3599*   3856*   4104*   4396*   4653*   4901*   5193*   5446*   5762*   6015*   6272*   6646*
                     6734*   6931*   7170*   7395*   7660*   7924*   8172*   8429*   8686*   9388*   9817   11147   12223
                    13730
RMR   = 000004       1002#  11548   11580   11584   11862   11880   11883   12168   12199   12202
RMSN  = 000030       1131#
RMSNI   001356       1397#  13730
RMSNO   001426       1424#
RMWC  = 000002       1208#   3468*   3469    3619    3688    3767    3876    3945    4023    4124    4193    4274    4416
                     4485    4564    4673    4742    4820    4921    4990    5071    5213    5287    5365    5466    5565
                     5672    5782    5856    5934    6035    6109    6187    6299    6368    6500    6673    6742    6833
                     6988    7093    7216    7308    7415    7489    7571    7680    7749    7829    7944    8013    8091
                     8192    8261    8340    8449    8518    8597    8706    8775    8854    9394
RMWCI   001330       1386#   9933   10882   10894   10912   10976   13730
RMWCO   001400       1413#   3598*   3681*   3855*   3938*   4103*   4185*   4395*   4478*   4652*   4735*   4900*   4982*
                     5192*   5254*   5445*   5557*   5761*   5823*   6014*   6077*   6271*   6361*   6493*   6645*   6930*
                     7169*   7394*   7480*   7659*   7741*   7822*   7923*   8005*   8171*   8253*   8428*   8510*   8685*
                     8768*   9387*   9649    9808    9934   10895   10913
RQA   = 100000       1087#  11457   11999
RQB   = 040000       1088#  11457   11999
RTC   = 000016        929#
SADMSK= 000037        971#
SAVREG= 104414       8994   13627#
SA1   = 000001        966#
```

```
SA16  = 000020            962#
SA2   = 000002            965#
SA4   = 000004            964#
SA8   = 000010            963#
SC    = 100000           1159#  10299  10316  10332  11385
SCTMSG  066704           3606   3863   4111   4403   4660   4908   5200   5453   5769   6022   6279   6653   7402
                         7667   7931   8179   8436   8693  13730#
SCTMSK= 003700           1022#
SC0   = 000100           1020#
SC1   = 000200           1019#
SC2   = 000400           1018#
SC3   = 001000           1017#
SC4   = 002000           1016#
SEARCH= 000030            935#  10475  10876
SECERR  044624           3730   3809   3987   4065   4235   4350   4527   4606   4784   4862   5032   5147   5329
                         5407   5604   5712   5898   5976   6151   6229   6463   6594   6797   6888   7037   7142
                         7262   7356   7531   7613   7791   7871   8055   8133   8303   8382   8560   8639   8817
                         8896  10395#
SEEK  = 000004            924#  5528   5636
SEKSTS  050700           5545   5653  11109#
SHUT    062644          12828#  13327  13410
SIZCLK  043464           3424  10142#
SKI   = 040000           1108#  11139  11142  11158  11188  11190  11194  11205  11267  11470  11722  11746  11749
                        11848   12213  12216  12230  12241  12243  12244  12246  12258  12550  12680  12695  12714
                        12717
SNGPRT= 020024           1070#  3559   3563
STACK = 001100            799#  3197   3457   3497   3541   3575   3832   4080   4372   4629   4877   5169   5422
                         5738   5991   6244   6620   6921   7159   7371   7636   7900   8148   8405   8662
STANDA  005772           3261   3272#
START   005322           1245   3181#  3486  12834
STCDRV  061676          12633#
STKLMT= 177774            810#
STOP    062614          12811#  13148
STSD1   104370          13730#
STSD2   104406          13730#
STSD3   104426          13730#
STSD4   104440          13730#
STSF    104471          13730#
STSH1   104033          13730#
STSH2   104110          13730#
STSH3   104177          13730#
STSH4   104235          13730#
SWR     001154           1298#  3195   3217*  3219   3225*  3232*  3246   3435   8995  13149  13163  13166  13181
                        13188   13250  13257  13269  13273  13330  13369  13424*  13641  13654*
SWREG   000176           1227#  3225   3246   3435  13330  13369  13392
SW0   = 000001            863#
SW00  = 000001            853#   863
SW01  = 000002            852#   862
SW02  = 000004            851#   861
SW03  = 000010            850#   860
SW04  = 000020            849#   859
SW05  = 000040            848#   858
SW06  = 000100            847#   857
SW07  = 000200            846#   856
SW08  = 000400            845#   855
SW09  = 001000            844#   854
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
SW1    = 000002            862#
SW10   = 002000            843#
SW11   = 004000            842#
SW12   = 010000            841#
SW13   = 020000            840#   8995
SW14   = 040000            839#
SW15   = 100000            838#
SW2    = 000004            861#
SW3    = 000010            860#
SW4    = 000020            859#
SW5    = 000040            858#
SW6    = 000100            857#
SW7    = 000200            856#
SW8    = 000400            855#
SW9    = 001000            854#
TADMSK= 003400             970#
TAG    = 020000           1089#
TAGADR= 001114            1270#   1277
TAP    = 040000           1066#
TA1    = 000400            961#
TA2    = 001000            960#
TA4    = 002000            959#
TBITVE= 000014             896#
TIMOUT  043606            3637    3703    3782    3894    3960    4038    4142    4208    4289    4434    4500    4579    4691
                          4757    4835    4939    5005    5086    5231    5302    5380    5484    5538    5577    5646    5685
                          5800    5871    5949    6053    6124    6202    6317    6383    6515    6691    6757    6848    7003
                          7108    7243    7338    7433    7504    7586    7698    7764    7844    7962    8028    8106    8210
                          8276    8355    8467    8533    8612    8724    8790    8869    9305    9483    9517   10171#
TKVEC  = 000060            903#  13307*  13308*
TPVEC  = 000064            904#
TRAPVE= 000034             902#    3203*   3204*
TRE    = 040000           1160#  10299   10316   10332   10477   10479   10884
TRTVEC= 000014             897#
TST    = 010000           1090#  11458   12000
TSTNMB  037212            9006*   9007*   9009   9126#
TSTPRP  037260            3586    3668    3749    3843    3925    4006    4091    4173    4257    4383    4465    4546    4640
                          4722    4803    4888    4970    5054    5180    5269    5348    5433    5518    5626    5749    5838
                          5917    6002    6091    6170    6259    6348    6481    6633    6722    6815    6950    7056    7188
                          7281    7382    7464    7550    7647    7729    7810    7911    7993    8074    8159    8241    8322
                          8416    8498    8579    8673    8755    8836    9174#
TSTQUE  001446            1437#   3407    3408*   3447    3459    3499    3543    3577    3834    4082    4374    4631    4879
                          5171    5424    5740    5993    6246    6622    6923    7161    7373    7638    7902    8150    8407
                          8664    8922    8924*   8928*  11057   11342   11409   11970
TST1    006774            3452#  13208
TST10   014016            4626#  13215
TST11   015010            4874#  13216
TST12   016252            5166#  13217
TST13   017272            5419#  13218
TST14   020620            5735#  13219
TST15   021640            5988#  13220
TST16   022660            6241#  13221
TST17   024370            6617#  13222
TST2    007172            3494#  13209
TST20   025654            6918#  13223
TST21   026762            7156#  13224
TST22   027730            7368#  13225
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
TST23   031006          7633#  13226
TST24   032074          7897#  13227
TST25   033066          8145#  13228
TST26   034110          8402#  13229
TST27   035132          8659#  13230
TST3    007356          3538#  13210
TST4    007520          3572#  13211
TST5    010542          3829#  13212
TST6    011534          4077#  13213
TST7    012774          4369#  13214
TYPBN = 104406          9116   13619#
TYPDS = 104405          8956   8963   9110   13618#
TYPE  = 104401          3240   3281   3284   3290   3295   3298   3301   3305   3308   3314   3317   3323   3328
                        3333   3339   3349   3354   3360   3373   3374   3377   3382   3385   3392   3606   3863
                        4111   4403   4660   4908   5200   5453   5769   6022   6279   6653   7402   7667   7931
                        8179   8436   8693   8950   8957   8964   8999   9000   9008   9017   9023   9032   9070
                        9076   9080   9093   9099   9101   9119   9121   12829  12901  12967  13036  13096  13252
                        13260  13324  13336  13390  13391  13394  13405  13415  13426  13445  13493  13499  13504
                        13508  13513  13514  13516  13519  13523  13614#  13663
TYPOC = 104402          3316   9113   13393  13615#
TYPON = 104404          13617#
TYPOS = 104403          3330   3351   9003   9011   9020   9026   13616#
UBUSQS  067061          3305   13730#
UNS   = 040000          990#   7199   7291   11425  11548  11552  11556  11862  11894  11899  12316  12319  12324
UNTMSK= 000007          1191#  10257  10259
UPE   = 020000          1173#  10641
USE   = 040000          1119#  6407   6538   9592   9815
USRFIL  105510          9606*  9613*  9764   9766   13730#
U0    = 000001          1188#  7477   7562
U1    = 000002          1187#  7477   7562
U2    = 000004          1186#  7477   7562
VV    = 000100          984#   9242   9437   10567  11179  11244  11245  11294  11299  11520  11523  11736  11775
                        11776  11799  11805  11946  12130  12159  12542  12543  12581  12586  12641  12642  12662
                        12667
WC    = 000040          1056#  11435  11987
WCD   = 000050          944#   4017   4268   4814   5065   5359   5928   6181   8085
WCE   = 040000          1172#  4307   4317   5104   5114   10540  13730
WCEHI = 010000          1197#
WCELO = 004000          1198#
WCF   = 000040          999#   1006   10611  12516  12519
WCH   = 000052          945#
WD    = 000060          948#   3682   3939   4187   4479   4736   4984   5256   5281   5559   5827   5850   6076
                        6103   6362   6735   6932   6981   7210   7482   7743   8007   8255   8512   8769
WH    = 000062          949#   3600   3857   4105   4397   4654   4902   5194   5447   5763   6016   6273   6647
                        7171   7396   7661   7925   8173   8430   8687   12379
WLE   = 004000          993#   1006   10592  10596  10609  10624  10639  12316  12348  12352  12360  13730
WRL   = 004000          979#   10594  12355
ZEROS   070060          3610   3867   4115   5204   5825   6026   6283   7671   13730#
$APTHD  001100          1257   1263#
$ASTAT= ****** U        13706  13721
$ATYC   066462          13677  13679#
$ATY1   066436          13675#
$ATY3   066444          13081  13676#
$ATY4   066454          13265  13678#
$AUTOB  001150          1295#  3250*  3439*  13383  13540
$BASE   001276          1364#  3315   3325*  3458   3498   3542   3576   3833   4081   4373   4630   4878   5170
```

```
                    5423   5739   5992   6245   6621   6922   7160   7372   7637   7901   8149   8406   8663
                   10029  10102  10179  11056  11340  13730
$BDADR  001136      1290#  4338#  4339#  4340   5135#  5136#  5137   9944#  13730
$BDDAT  001142      1292#  4318#  4334#  4344   5115#  5131#  5141   6422#  6442#  6458#  6553#  6573#  6589#
                    6777#  6778#  6782#  6783#  6868#  6869#  6873#  6874#  7031#  7137#  7238#  7332#  9942#
                   10256#  10257#  10260  10278#  10284#  10300#  10317#  10333#  10361#  10411#  10412#  10424#  10425#
                   10435#  10436#  10438  10456#  10457#  10458  10478#  10479#  10480  10511#  10512#  10513  10527#
                   10528#  10529  10541#  10542#  10554#  10555#  10571#  10572#  10597#  10598#  10612#  10613#  10627#
                   10628#  10642#  10643#  10655#  10656#  10685#  10686#  10700#  10701#  10715#  10716#  10732#  10733#
                   10758#  10759#  10773#  10774#  10788#  10789#  10803#  10804#  10826#  10827#  10841#  10842#  10863#
                   10864#  10882#  10903#  10954#  10955#  10956  10981#  10982#  10983  10997#  10998  11009#  11010
                   11127#  11154#  11155#  11156  11176#  11187#  11188#  11194  11205  11218#  11231#  11256#  11270#
                   11271#  11284#  11300#  11384#  11385#  11387  11397#  11406#  11414  11424#  11425#  11434#  11435#
                   11437  11447#  11456#  11457#  11459  11469#  11470#  11479#  11480#  11482  11524#  11538#  11554#
                   11568#  11582#  11596#  11610#  11665#  11679#  11695#  11712#  11733#  11750#  11764#  11789#  11804#
                   11819#  11835#  11852#  11870#  11884#  11900#  11933#  11934#  11936  11945#  11946#  11948  11958#
                   11967#  11974#  11977  11986#  11987#  11989  11998#  11999#  12001  12011#  12020#  12070#  12086#
                   12107#  12127#  12156#  12175#  12189#  12203#  12226#  12227#  12228  12240#  12241#  12277#  12293#
                   12307#  12325#  12338#  12350#  12393#  12406#  12419#  12432#  12454#  12479#  12494#  12507#  12520#
                   12533#  12554#  12571#  12587#  12652#  12668#  12684#  12704#  12718#  12774#  12789#  13730
$BELL   001212      1315#  13252  13285  13336  13533
$BIN    063060     12897#  12900#  12901  12908#
$CDW1   001302      1366#
$CDW2   001304      1367#
$CHARC  064012     13098#  13108#  13115  13124#  13129#
$CKSWR  065110     13369#  13623
$CMTAG  001114      1278#  3192   3193   3201   3207   3208   3209
$CM3    =  000000   1308#
$CM4    =  000005   1308#  1309#  1310#  1311#  1312#  1313#
$CNTLC  066006     13324  13405  13533#
$CNTLG  066020     13390  13535#
$CNTLU  066013     13415  13508  13534#
$CPUOP  001250      1338#
$CRLF   001217      1317#  8964   8999   9032   9070   9080   9093   9101   9121   13097  13132  13260  13285
                   13426  13513  13533
$DBLK   063276     12933  12967  12975#
$DDW0   001306      1368#
$DDW1   001310      1369#
$DDW2   001312      1370#
$DDW3   001314      1371#
$DDW4   001316      1372#
$DDW5   001320      1373#
$DDW6   001322      1374#
$DDW7   001324      1375#
$DEVCT  001232      1329#
$DEVM   001300      1365#  3267#  3372#  3380#  3398#  3399   3406
$DOAGN  036446      8946   8967   8973#
$DTBL   063266     12936  12971#
$ENDAD  036436      1236   3238   8969#  13280
$ENDCT  036302      3207   8948#
$ENULL  036452      8976#
$ENV    001242      1334#  3244   3260   3433   13076  13262  13684  13708
$ENVM   001243      1335#  3230   3265   13078  13083  13686
$EOP    036246      3487   8938#  12832
$EOPCT  036274      3207#  8945#  8949
$EOSP   036212      3530   3566   8920#
```

```
SERFLG  001117          1281#  13137  13177  13179  13185*  13207  13247*  13285
SERMAX  001131          1287#   3210* 13179  13202* 13207
SERROR  064412          3201   13245#
SERRPC  001132          1288#   9024  13254* 13255* 13256   13285
SERRTB  001562          1476#   9039
SERTTL  001126          1285#   8961   8965* 13253* 13285
SESCAP  001210          1314#   3209*  4246*  4363*  5043*  5160*  5507*  5552*  5616*  5660*  13201* 13276  13278
                       13285
SETABL  001242          1333#
SETEND  001326          1269   1376#
SFATAL  001224          1326#  13712#
SFFLG   066702         13675*  13678* 13706  13715* 13723#
SFILLC  001172          1306#  13101  13132
SFILLS  001171          1305#  13132
SGDADR  001134          1289#   4337*  4340   5134*  5137   9943*  13730
SGDDAT  001140          1291#   4316*  4317*  4335*  4336*  4344    5113*  5114*  5132*  5133*  5141   6423*  6441*
                        6459*   6554*  6572*  6590*  6784*  6875*   7032*  7033*  7135*  7136*  7236*  7237*  7330*
                        7331*   9941*  10258* 10259* 10260  10276*  10277*  10283*  10285*  10297*  10298*  10299*  10315*
                       10316*  10331*  10332* 10359* 10360* 10413*  10426*  10437*  10438   10450*  10455*  10458   10469*
                       10477*  10480   10506* 10509* 10510* 10513   10525*  10526*  10529   10536*  10888*  10894*  10895*
                       10896*  10897*  10900* 10901  10948* 10953*  10956   10965*  10978*  10983   10994*  10995*  10996*
                       10998   11007*  11008* 11010  11126* 11138*  11151*  11156   11175*  11189*  11190*  11217*  11230*
                       11254*  11255*  11269* 11282* 11283* 11298*  11299*  11386*  11387   11396*  11412*  11414   11423*
                       11436*  11437   11446* 11458* 11459  11468*  11481*  11482   11522*  11523*  11536*  11537*  11555*
                       11556*  11569*  11570* 11583* 11584* 11597*  11598*  11611*  11612*  11663*  11664*  11677*  11678*
                       11693*  11694*  11710* 11711* 11731* 11732*  11748*  11749*  11762*  11763*  11787*  11788*  11803*
                       11805*  11817*  11818* 11833* 11834* 11850*  11851*  11868*  11869*  11882*  11883*  11898*  11899*
                       11935*  11936   11947* 11948  11957* 11988*  11989   12000*  12001   12009*  12068*  12069*  12084*
                       12085*  12105*  12106* 12125* 12126* 12154*  12155*  12173*  12174*  12187*  12188*  12201*  12202*
                       12212*  12225*  12228  12242* 12243* 12275*  12276*  12291*  12292*  12305*  12306*  12323*  12324*
                       12336*  12337*  12351* 12352* 12360* 12391*  12392*  12404*  12405*  12417*  12418*  12430*  12431*
                       12452*  12453*  12477* 12478* 12492* 12493*  12505*  12506*  12518*  12519*  12531*  12532*  12552*
                       12553*  12569*  12570* 12585* 12586* 12650*  12651*  12666*  12682*  12683*  12702*  12703*  12716*
                       12717*  12775*  12790* 13730
SGET42  036426          8966#
SGTSWR  065200         13391#  13621
SHD     =  000001        775     776
SHIBTS  001100          1264#
SHIOCT  066150         13573*  13578#
SICNT   001120          1282#  13192*  13193  13195*  13206
SILLUP  066420         13633   13649   13666#
SINTAG  001151          1296#  13388*  13407  13427   13540
SITEMB  001130          1286#   9015   13256* 13264   13285
SLF     001220          1318#  13132   13285  13523   13533
SLFLG   066701         13716*  13722#
SLPADR  001122          1283#   3211*  3454*  3657*  3739*  3914*  3996*  4162*  4244*  4364*  4454*  4536*  4711*
                        4793*   4959*  5041*  5161*  5251*  5338*  5504*  5613*  5820*  5907*  6073*  6160*  6337*
                        6472*   6612*  6711*  6806*  6913*  6939*  7046*  7151*  7177*  7271*  7453*  7540*  7718*
                        7800*   7892*  7982*  8064*  8230*  8312*  8487*  8569*  8744*  8826*  8917*  13174*  13183*
                       13199*  13204  13206
SLPERR  001124          1284#   3212*  3455*  3658*  3740*  3915*  3997*  4163*  4245*  4365*  4455*  4537*  4712*
                        4794*   4960*  5042*  5162*  5252*  5339*  5505*  5614*  5821*  5908*  6074*  6161*  6338*
                        6473*   6613*  6712*  6807*  6914*  6940*  7047*  7152*  7178*  7272*  7454*  7541*  7719*
                        7801*   7893*  7983*  8065*  8231*  8313*  8488*  8570*  8745*  8827*  8918*  13183   13200*
                       13206   13275
SMADR1  001254          1351#
```

```
$MADR2  001260          1355#
$MADR3  001264          1358#
$MADR4  001270          1361#
$MAIL   001222          1265   1269   1324#  3229   3244   3433   3460   3500   3544   3578   3835   4083   4375
                        4632   4880   5172   5425   5741   5994   6247   6623   6924   7162   7374   7639   7903
                        8151   8408   8665   13076  13198  13262
$MAMS1  001252          1345#
$MAMS2  001256          1353#
$MAMS3  001262          1356#
$MAMS4  001266          1359#
$MBADR  001102          1265#
$MFLG   066700         13676#  13682  13717#  13721#
$MNEW   066036         13394   13538#
$MSGAD  001236          1331#  13692#  13695
$MSGLG  001240          1332#  13697#
$MSGTY  001222          1325#  13690  13698#  13710  13714#
$MSWR   066025         13391   13536#
$MTYP1  001253          1346#
$MTYP2  001257          1354#
$MTYP3  001263          1357#
$MTYP4  001267          1360#
$MXCNT  064332         13196   13206#
$NULL   001170          1304#  13103  13132
$NWTST= 000001          3449#   3451   3490#  3492   3534#  3536   3569#  3826#  4074#  4366#  4623#  4871#  5163#
                        5416#   5732#  5985#  6238#  6614#  6915#  7153#  7365#  7630#  7894#  8142#  8399#  8656#
$OCNT   063530         13008#  13037#  13050#
$OMODE  063532         13003#  13007#  13012  13015#  13026#  13052#
$OVER   064316         13150   13175   13184  13194   13203#
$PASS   001230          1328#   3229#  8942#  8943#  8954   8976   13190  13207
$PASTM  001106          1267#
$POWER  066426         13664   13669#
$PWRON  066260          3205   13633#  13661
$PWRMG  066414         13664#
$PWRUP  066332         13643   13649#
$QUES   001216          1316#  13132  13285  13445  13516  13533
$RDCHR  065452         13458#  13624
$RDDEC= ****** U       13627
$RDLIN  065542         13481#  13625
$RDOCT  066050         13551#  13626
$RDSZ = 000010         13474#
$RESRE  062750         12873#  13628
$RTNAD  036450          8975#
$R2A  = ****** U       13629
$SAVRE  062712         12857#  13627
$SAVR6  066424         13642#  13650  13651#  13652#  13668#
$SCOPE  064016          3199   13146#
$SETUP= 000137          1218#   3198   3199   3201   3203   3205   3207   3208   3209   3211   3238   3241   3430
                        8940   13147  13246  13272  13280  13328  13333  13334  13364  13540
$STUP = 177777          1218#
$SVLAD  064262         13158   13197#
$SVPC = 000200          1234#   1239
$SWR  = 167400           764#   775    780    781    782    783    784    785    786    787    1313   1314   1315
                        3208   3209   3211   3212   3460   3500   3544   3578   3835   4083   4375   4632   4880
                        5172   5425   5741   5994   6247   6623   6924   7162   7374   7639   7903   8151   8408
                        8665   8935   8941   8968   8974   8976   13138  13139  13140  13141  13142  13149  13161
                       13163  13164  13177  13178  13179  13186  13187  13188  13200  13203  13206  13237  13238
```

```
                    13239   13240   13241   13250   13257   13269   13273   13285   13665
$SWREG  001244       1336#   3232
$SWRMK= 000000        787     788    13142   13143   13167
$SWOBT  064334      13173   13207#
$TESTN  001226       1327#   3460#   3500#   3544#   3578#   3835#   4083#   4375#   4632#   4880#   5172#   5425#   5741#
                     5994#   6247#   6623#   6924#   7162#   7374#   7639#   7903#   8151#   8408#   8665#   9007   13198#
$TIMES  001206       1313#   3208#   8941#  13186#  13193   13196#  13206
$TKB    001162       1301#  13288   13309   13320   13345   13373   13400
$TKCNT  064612      13289#  13304#  13334   13351#  13465   13467#
$TKINT  064622        3273    3442  13304#  13325   13386
$TKQEN= 064621      13293#  13359   13470
$TKQIN  064614      13290#  13305#  13306   13357#  13358#  13359   13361#
$TKQOU  064616      13291#  13306#  13468   13469#  13470   13472#
$TKQSR  064620      13292#  13305   13361   13472
$TKS    001160       1300#  13288   13310#  13341#  13343   13349#  13371   13387#  13397   13409#  13429#
$TKSRV  064672      13307   13320#
$TMPO   001174       1308#   3514#   3517    3610#   3867#   4115#   4319#   4407#   4664#   4912#   5116#   5204#   5257#
                     5457#   5773#   5825#   6026#   6079#   6283#   6424#   6555#   6657#   7406#   7671#   7935#   8183#
                     8440#   8697#   9838    9945#  10034#  10107#  11062#  11063   13730
$TMP1   001176       1309#   3283#   3284    3285    3288    3297#   3298    3299    3307#   3308    3309    3319#   3321
                     3325    3326#   3327#   3329    3335#   3337    3341    3342#   3343#   3344#   3345#   3346#   3347#
                     3348#   3350    3356#   3358    3363#   3364#   3365#   3366#   3367#   3368    3376#   3377    3378
                     3384#   3385    3386    3388    3390    3394    3513#   3521    3563#   3611#   3868#   4116#   4320#
                     4408#   4665#   4913#   5117#   5205#   5258#   5458#   5774#   5826#   6027#   6080#   6284#   6425#
                     6556#   6658#   7407#   7672#   7936#   8184#   8441#   8698#   9839   10035#  10036   10108#  10109
                    11061#  11068   12759#  12760#  12772   12774   13730
$TMP2   001200       1310#   3564#  12762#  12763#  12786   12789   13730
$TMP3   001202       1311#
$TMP4   001204       1312#
$TN   = 000030        775#   3449    3460#   3490    3500#   3534    3544#   3569    3578#   3826    3835#   4074    4083#
                     4366    4375#   4623    4632#   4871    4880#   5163    5172#   5416    5425#   5732    5741#   5985
                     5994#   6238    6247#   6614    6623#   6915    6924#   7153    7162#   7365    7374#   7630    7639#
                     7894    7903#   8142    8151#   8399    8408#   8656    8665#  13168   13208
$TPB    001166       1303#  13121#  13132
$TPFLG  001173       1307#  13070   13132
$TPS    001164       1302#  13119   13132
$TRAP   066152        3203  13587#
$TRAP2  066212      13602#  13613
$TRP  = 000016      13606#  13615#  13616#  13617#  13618#  13619#  13620#  13621   13622#  13623   13624#  13625#  13626#
                    13627#  13628#  13629#
$TRPAD  066224      13596   13613#
$TSTM   001104       1266#
$TSTNM  001116       1280#   8940#  13137   13170#  13197#  13198   13203   13207   13249   13285
$TTYIN  065776      13483   13484   13496   13514   13528   13532#
$TYPBN  063006      12894#  13619
$TYPDS  063062      12921#  13618
$TYPE   063534      13070#  13606   13614   13703
$TYPEC  063746      13100   13107   13114   13119#  13120   13431
$TYPEX  064014      13125   13127   13130#
$TYPOC  063332      13006#  13615
$TYPON  063346      13005   13008#  13617
$TYPOS  063306      13001#  13616
$UNIT   001234       1330#   3447*   9001   10258
$UNITM  001110       1268#
$USWR   001246       1337#
$VECT1  001272       1362#   3326    3341*   3342    3368*
```

DZRMEA.P11 29-JUL-77 14:18 CROSS REFERENCE TABLE -- USER SYMBOLS

```
$VECT2  001274            1363#
$XTSTR  064034           13152#
$$GET4= 000000            8968#
$$SW08= 000030           13207#  13208   13209#  13210#  13211#  13212#  13213#  13214#  13215#  13216#  13217#  13218#  13219#
                         13220#  13221#  13222#  13223#  13224#  13225#  13226#  13227#  13228#  13229#  13230#  13231#
$OFILL  063531           13002*  13006*  13016   13051#
$40CAT= ****** U         13149   13259
.       = 111113          1221#   1225#   1234    1235#   1237#   1239#   1244#   1247#   1253    1254#   1256#   1258#   1270
                          1277#   1319    1437#   1453#   1458#   3196    3211    3212    3254#   8953#   8976    8977#  12839#
                         12975#  13132   13206   13207   13285   13288   13292#  13293   13294#  13532#  13533   13540#  13645
                         13667   13724#  13730#
.$ASTA= ****** U         13676   13679
.$X     = 001100          1253#   1258
```

CALCLR 764# 3502 3546
CALGET 764#
CALPUT 764#
CALSUB 764# 3586 3604 3626 3634 3637 3640 3648 3668 3692 3700 3703 3706 3714 3721
3730 3749 3771 3779 3782 3785 3793 3800 3809 3816 3843 3861 3883 3891 3894
3897 3905 3925 3949 3957 3960 3963 3971 3978 3987 4006 4027 4035 4038 4041
4049 4056 4065 4091 4109 4131 4139 4142 4145 4153 4173 4197 4205 4208 4211
4219 4226 4235 4257 4278 4286 4289 4292 4300 4309 4350 4383 4401 4423 4431
4434 4437 4445 4465 4489 4497 4500 4503 4511 4518 4527 4546 4568 4576 4579
4582 4590 4597 4606 4613 4640 4658 4680 4688 4691 4694 4702 4722 4746 4754
4757 4760 4768 4775 4784 4803 4824 4832 4835 4838 4846 4853 4862 4888 4906
4928 4936 4939 4942 4950 4970 4994 5002 5005 5008 5016 5023 5032 5054 5075
5083 5086 5089 5097 5106 5147 5180 5198 5220 5228 5231 5234 5242 5269 5291
5299 5302 5305 5313 5320 5329 5348 5369 5377 5380 5383 5391 5398 5407 5433
5451 5473 5481 5484 5487 5495 5518 5529 5537 5538 5539 5545 5569 5577 5580
5588 5595 5604 5626 5637 5645 5646 5647 5653 5676 5685 5688 5696 5703 5712
5719 5749 5767 5789 5797 5800 5803 5811 5838 5860 5868 5871 5874 5882 5889
5898 5917 5938 5946 5949 5952 5960 5967 5976 6002 6020 6042 6050 6053 6056
6064 6091 6113 6121 6124 6127 6135 6142 6151 6170 6191 6199 6202 6205 6213
6220 6229 6259 6277 6306 6314 6317 6320 6328 6348 6372 6380 6383 6386 6394
6415 6434 6451 6463 6481 6504 6512 6515 6518 6526 6546 6565 6582 6594 6633
6651 6680 6688 6691 6694 6702 6722 6746 6754 6757 6760 6768 6788 6797 6815
6837 6845 6848 6851 6859 6879 6888 6895 6935 6950 6992 7000 7003 7006 7014
7023 7037 7056 7097 7105 7108 7111 7119 7128 7142 7174 7188 7222 7223 7229
7243 7246 7254 7262 7281 7313 7315 7323 7338 7341 7349 7356 7382 7400 7422
7430 7433 7436 7444 7464 7493 7501 7504 7507 7515 7522 7531 7550 7575 7583
7586 7589 7597 7604 7613 7620 7647 7665 7687 7695 7698 7701 7709 7729 7753
7761 7764 7767 7775 7782 7791 7810 7833 7841 7844 7847 7855 7862 7871 7878
7911 7929 7951 7959 7962 7965 7973 7993 8017 8025 8028 8031 8039 8046 8055
8074 8095 8103 8106 8109 8117 8124 8133 8159 8177 8199 8207 8210 8213 8221
8241 8265 8273 8276 8279 8287 8294 8303 8322 8344 8352 8355 8358 8366 8373
8382 8389 8416 8434 8456 8464 8467 8470 8478 8498 8522 8530 8533 8536 8544
8551 8560 8579 8601 8609 8612 8615 8623 8630 8639 8646 8673 8691 8713 8721
8724 8727 8735 8755 8779 8787 8790 8793 8801 8808 8817 8836 8858 8866 8869
8872 8880 8887 8896 8903
CNACMS 3449# 3451
COMMEN 906#
DVAVMS 3490# 3492
DVTYMS 3534# 3536
ENDCOM 906#
ERROR 800# 3426 3483 3505 3519 3523 3529 3549 3558 3565 3590 3607 3629 3643 3651
3672 3695 3709 3717 3724 3733 3753 3774 3788 3796 3803 3812 3821 3847 3864
3886 3900 3908 3929 3952 3966 3974 3981 3990 4010 4030 4044 4052 4059 4068
4095 4112 4134 4148 4156 4177 4200 4214 4222 4229 4238 4261 4281 4295 4303
4312 4321 4330 4342 4346 4353 4387 4404 4426 4440 4448 4469 4492 4506 4514
4521 4530 4550 4571 4585 4593 4600 4609 4618 4644 4661 4683 4697 4705 4726
4749 4763 4771 4778 4787 4807 4827 4841 4849 4856 4865 4892 4909 4931 4945
4953 4974 4997 5011 5019 5026 5035 5058 5078 5092 5100 5109 5118 5128 5139
5143 5150 5184 5201 5223 5237 5245 5273 5294 5308 5316 5323 5332 5352 5372
5386 5394 5401 5410 5437 5454 5476 5490 5498 5522 5532 5542 5548 5572 5583
5591 5598 5607 5630 5640 5650 5656 5679 5691 5699 5706 5715 5724 5753 5770
5792 5806 5814 5842 5863 5877 5885 5892 5901 5921 5941 5955 5963 5970 5979
6006 6023 6045 6059 6067 6095 6116 6130 6138 6145 6154 6174 6194 6208 6216
6223 6232 6263 6280 6309 6323 6331 6352 6375 6389 6397 6418 6426 6437 6443
6454 6460 6466 6485 6507 6521 6529 6549 6557 6568 6574 6585 6591 6597 6637
6654 6683 6697 6705 6726 6749 6763 6771 6785 6791 6800 6819 6840 6854 6862

```
            6876    6882    6891    6900    6954    6965    6974    6995    7009    7017    7026    7034    7040    7060    7071
            7080    7100    7114    7122    7131    7138    7145    7192    7203    7226    7232    7239    7249    7257    7265
            7285    7295    7318    7326    7333    7344    7352    7359    7386    7403    7425    7439    7447    7468    7496
            7510    7518    7525    7534    7554    7578    7592    7600    7607    7616    7625    7651    7668    7690    7704
            7712    7733    7756    7770    7778    7785    7794    7814    7836    7850    7858    7865    7874    7883    7915
            7932    7954    7968    7976    7997    8020    8034    8042    8049    8058    8078    8098    8112    8120    8127
            8136    8163    8180    8202    8216    8224    8245    8268    8282    8290    8297    8306    8326    8347    8361
            8369    8376    8385    8394    8420    8437    8459    8473    8481    8502    8525    8539    8547    8554    8563
            8583    8604    8618    8626    8633    8642    8651    8677    8694    8716    8730    8738    8759    8782    8796
            8804    8811    8820    8840    8861    8875    8883    8890    8899    8908
ESCAPE       906#   4246    4362    5043    5159    5507    5552    5616    5660
GETCS1       764#
GETDB        764#   4325    5122
GETDS        764#
GETDT        764#   3552
GETMR1       764#
GETPRI       906#
GETSWR       764#    906#   3241#   3430
MULT         906#
NEWTST       906#   3449    3490    3534    3569    3826    4074    4366    4623    4871    5163    5416    5732    5985    6238
            6614    6915    7153    7365    7630    7894    8142    8399    8656
NWTST         764#  3452    3494    3538    3572    3829    4077    4369    4626    4874    5166    5419    5735    5988    6241
            6617    6918    7156    7368    7633    7897    8145    8402    8659
POP          906#   3476    3477    3481    3482    3515    3516    3527    3528    9399    9618    9619    9632    9775    9776
            9777    9849    9851    9852    9853    9854    9955    9957    9958    9959    9988    9989    9990   10063   10065
           10066   10067   10068   10069   10070   10130   10132   10133   10134   10135   10136   10137   10159   10161   10201
           10203   10204   10205   10206   10354   10358   10502   11086   11087   11088   11089   11349   11351   11352   11353
           11413   11975   11976   12878   12962   13574   13654   13655   13718   13719
PUSH         906#   3462    3463    3507    3509    9376    9601    9602    9625    9643    9644    9645    9801    9803    9804
            9805    9806    9877    9878    9879    9880    9972    9974    9975   10022   10023   10024   10025   10026   10027
           10028   10095   10096   10097   10098   10099   10100   10101   10143   10144   10172   10173   10174   10175   10176
           10350   10499   11050   11051   11052   11053   11333   11335   11336   11337   11407   11968   11969   12858   12921
           13553   13635   13641   13679   13681   13702
PUTCS1       764#
PUTDC        764#
PUTER1       764#   7197    7289
PUTMR1       764#   6959    6968    7065    7074
REPORT       906#
RGBFMC       764#   1383    1410
SCOPE        801#   3495    3539    3573    3830    4078    4370    4627    4875    5167    5420    5736    5989    6242    6618
            6919    7157    7369    7634    7898    8146    8403    8660
SETPRI       906#   3186    3443   12814   12820   13461
SETTRA     13606#  13615   13616   13617   13618   13619   13621   13623   13624   13625   13626   13627   13628
SETUP        906#   3191
SKIP         906#
SLASH        906#
SPACE        906#
STARS        906#   1232    1250    1252    1259    1273    1319    1322    3449    3451    3490    3493    3534    3537    3569
            3571    3580    3661    3743    3826    3828    3837    3918    4000    4074    4076    4085    4166    4248    4366
            4368    4377    4458    4540    4623    4625    4634    4715    4797    4871    4873    4882    4963    5045    5163
            5165    5174    5262    5342    5416    5418    5427    5511    5620    5732    5734    5743    5831    5911    5985
            5987    5996    6084    6164    6238    6240    6249    6253    6341    6475    6607    6614    6616    6625    6715
            6809    6915    6917    6943    7050    7153    7155    7181    7275    7365    7367    7376    7457    7544    7630
            7632    7641    7722    7804    7894    7896    7905    7986    8068    8142    8144    8153    8234    8316    8399
            8401    8410    8491    8573    8656    8658    8667    8748    8830    8931    8980    9181    9194    9206    9228
            9257    9279    9307    9371    9637   10397   10405   10490   10873   12842   12887   12911   12978   13055   13134
```

```
        13233   13287   13364   13379   13450   13474   13543   13581   13631   13647   13674
SWRSU     906#  3213#
TAGS      764#  1378
TRMTRP  13606#
TYPBIN    906#
TYPDEC    906#  8954    8961
TYPNAM    764#   906#   3234
TYPNUM    906#
TYPOCS    906#  3329    3350    9001    9009    9018    9024
TYPOCT    906#  3315   13392
TYPTXT    906#  8950    8957
XPER      764#  1480    1487    1494    1501    1508    1515    1522    1529    1536    1543    1550    1557    1564    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    1775    1782
         1789   1796    1803    1811    1819    1827    1834    1841    1848    1855    1862    1869    1876    1883    1891
         1898   1905    1912    1919    1926    1933    1940    1947    1954    1961    1968    1975    1982    1989    1996
         2003   2010    2017    2024    2031    2038    2045    2052    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    2284    2291    2298    2305    2312
         2319   2326    2333    2340    2347    2354    2361    2368    2375    2382    2389    2396    2403    2410    2417
         2424   2431    2438    2445    2452    2459    2466    2473    2480    2487    2494    2501    2508    2515    2522
         2529   2536    2543    2550    2557    2564    2571    2578    2585    2592    2599    2606    2613    2620    2627
         2634   2641    2648    2655    2662    2669    2676    2683    2690    2697    2704    2711    2718    2725    2732
         2739   2746    2753    2761    2768    2776    2783    2790    2798    2806    2814    2823    2831    2839    2846
         2853   2860    2867    2874    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    3084    3091    3098    3105    3112    3119    3126    3133    3140
$$CMRE   1271#
$$CMTM   1271#  1308    1309    1310    1311    1312
$$ESCA    906#
$$NEWT    906#  3449    3490    3534    3569    3826    4074    4366    4623    4871    5163    5416    5732    5985    6238
         6614   6915    7153    7365    7630    7894    8142    8399    8656
$$SET   13606#  13615   13616   13617   13618   13619   13621   13623   13624   13625   13626   13627   13628
$$SETM   3229#
$$SKIP    906#
.EQUAT    764#   796
.HEADE    764#   765
.SETUP    764#  1218
.SWRHI    764#   776
.SWRLO    764#   788#
.$ACTI    764#  1230
.$APTB    764#  1320#
.$APTH    764#  1248
.$APTY    764#  13672
.$CATC    764#  1219
.$CMTA    764#  1271
.$DIV     764#
.$EOP     764#  8929
.$ERRO    764#  13231
.$ERRT    764#
.$MULT    764#
.$POWE    764#  13629
.$RAND    764#
.$RDDE    764#
.$RDOC    764#  13541
.$READ    764#  13285
```

```
.$SAVE    764#  12840
.$SCOP    764#  13132
.$SIZE    764#
.$TRAP    764#  13579
.$TYPB    764#  12885
.$TYPD    764#  12909
.$TYPE    764#  13053
.$TYPO    764#  12976


. ABS.  111113     000


 ERRORS DETECTED:  0

 DSKZ:DZRMEA.BIN,DSKZ:DZRMEA.SEQ/DOC/SOL/CRF=DSKM:DZRMEA.P11
 RUN-TIME: 99 96 4 SECONDS
 RUN-TIME RATIO: 582/201=2.8
 CORE USED:  33K  (65 PAGES)

 DOCUMENT PAGES:  332
```