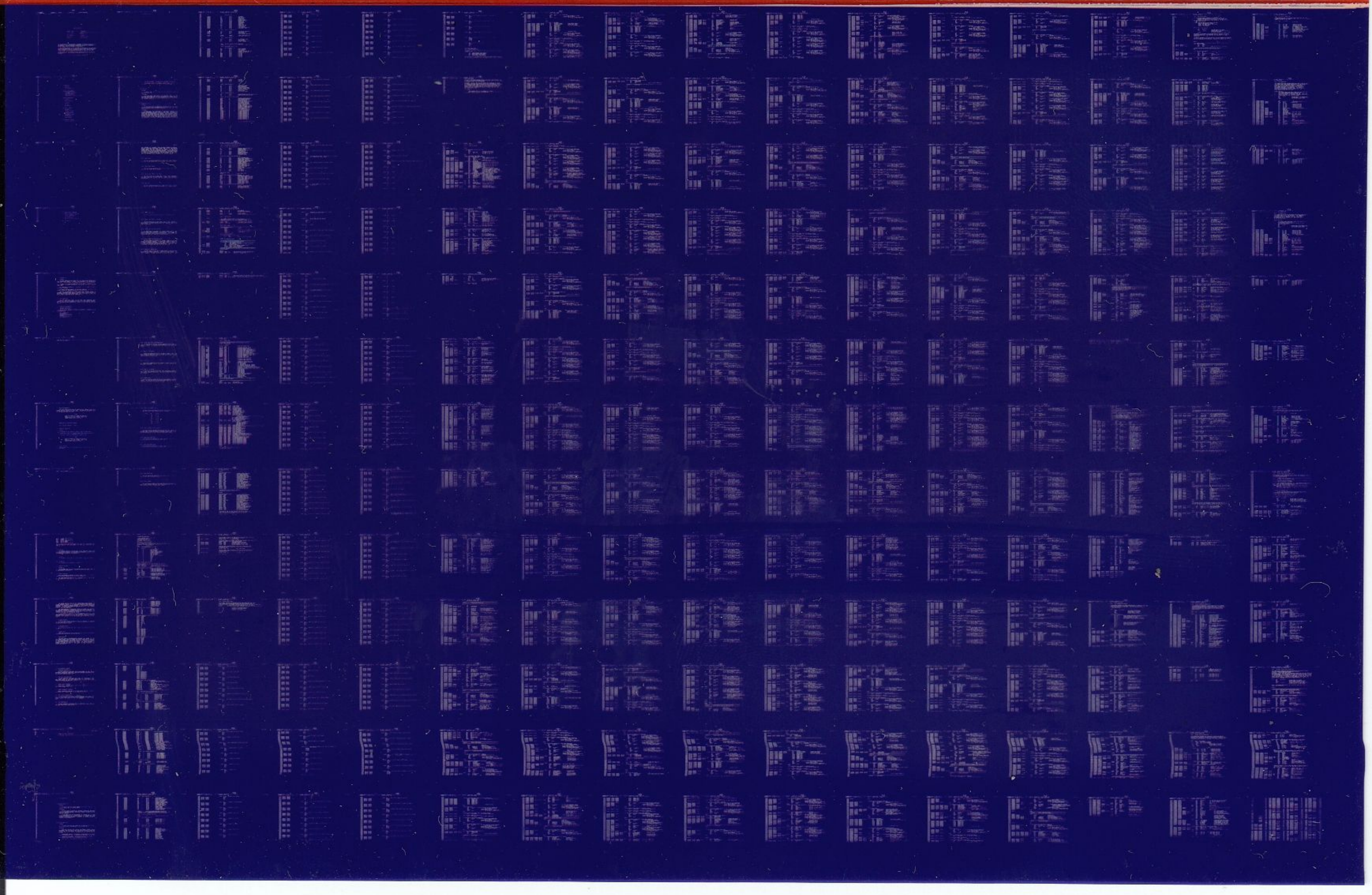
RM03,02

FUNCTIONAL TEST 3
CZRMEB0

AH-B004B-MC

FICHE 1 OF 2

JAN 1978
digital
MADE IN USA



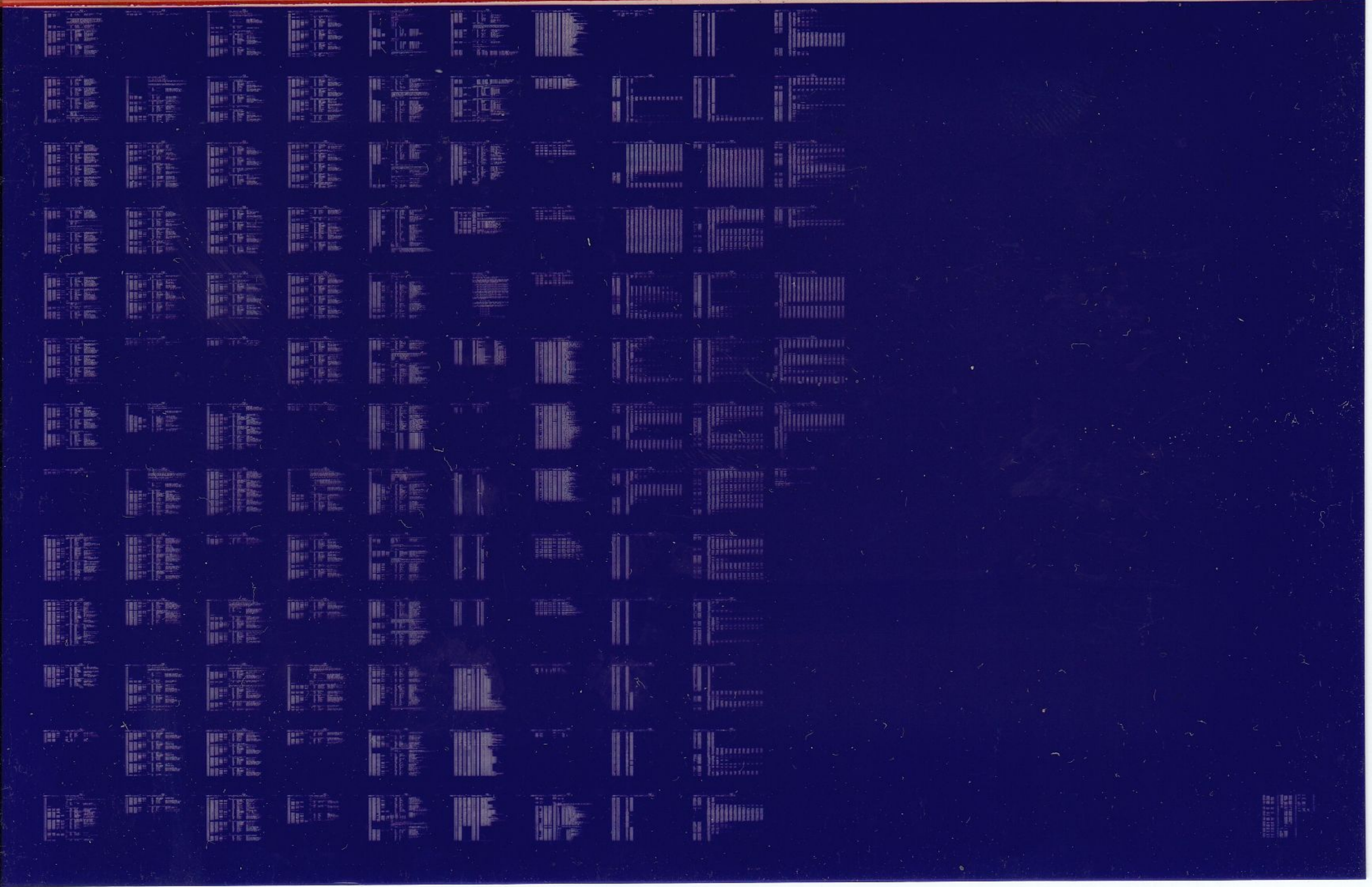
RM03,02

FUNCTIONAL TEST 3
CZRMEB0

AH-B004B-MC

FICHE 2 OF 2

JAN 1978
digital
MADE IN USA



```
                        .REM    \
```

IDENTIFICATION

PRODUCT CODE:	AC-B003B-MC
PRODUCT NAME:	CZRMEB0 RM03/RM02 FUNCTIONAL TEST, PART 3
DATE CREATED:	1 AUGUST 77
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	DOUG RIIKONEN

.PAGE

CONTENTS

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.

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.

```
182
183                     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, CZRMJ-B

3.0 OPERATING PROCEDURE

3.1 LOADING

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

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

.XXDP MEDIA, USING THE APPROPRIATE LOADING DEVICE.

NOTE

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

3.2 SWITCH OPTIONS

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

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

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

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

3.3 STARTING

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

3.4 HALTING

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

3.5 RESTARTING

THE PROGRAM CAN BE RESTARTED AT ADDRESS 200.

4.0 OPERATOR INTERFACE

4.1 PROGRAM ID

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

4.2 CONSOLE DIALOGUE

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

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

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

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

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

4.3 PROGRESS REPORTS

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

4.4 PERFORMANCE REPORT

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

4.5 PROGRAM HALTS

THERE ARE NO SCHEDULED HALTS DURING THE EXECUTION OF THE PROGRAM. 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

```
467                                     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

628 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.

```
736
737
738
739                     WRITE, READ EACH TRACK
740
741
742                          THE TEST WRITES AND READS WITH EACH  HEAD  USING  A  MIX  OF
743     EARLY, NORMAL AND LATE WRITE GATES.
744
745
746
747
748
749                     WRITE, READ W/ ABORT
750
751
752                          THE TEST SETS AN ERROR IN THE ERROR REGISTER AND EXECUTES  A
753     WRITE  DATA  COMMAND  VERIFYING THAT "PIP" REMAINS INACTIVE.  THE
754     SAME PROCEDURE IS USED FOR READ DATA COMMAND.
755
```

```
                        ;PROGRAM REVISION #001

                        .TITLE  CZRMEB0 RM03/2 FCTNL TST 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
```

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

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

                        .SBTTL  RM03 REGISTER BIT DEFINITIONS

                        ;RMCS1  CONTROL STATUS REGISTER

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

                        ;FUNCTION CODES (BITS 01-05 OF RMCS1)

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

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

                ;RMDA   DISK ADDRESS REGISTER

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

                ;TRACK,SECTOR MASKS

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

                ;RMDS   DRIVE STATUS REGISTER

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

```
                        ;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
```

```
1036
1037                            ;       READ ONLY BITS
1038
1039    100000                  OCC     =       BIT15           ;OCCUPIED
1040    040000                  RG      =       BIT14           ;RUN AND GO
1041    020000                  EBL     =       BIT13           ;END OF BLOCK
1042    010000                  REX     =       BIT12           ;EXCEPTION
1043    004000                  ESRC    =       BIT11           ;ENABLE SEARCH
1044    002000                  PLFS    =       BIT10           ;LOOKING FOR SYNC
1045    001000                  ECRC    =       BIT09           ;ENABLE CRC OUT
1046    000400                  PDA     =       BIT08           ;DATA AREA
1047    000200                  PHA     =       BIT07           ;HEADER AREA
1048    000100                  CONT    =       BIT06           ;CONTINUE
1049    000040                  WC      =       BIT05           ;WORD CLOCK
1050    000020                  EECC    =       BIT04           ;ENABLE ECC OUT
1051    000010                  MWD     =       BIT03           ;WRITE DATA BIT
1052    000004                  LS      =       BIT02           ;LAST SECTOR
1053    000002                  LST     =       BIT01           ;LAST SECTOR AND TRACK
1054    000001                  DMD     =       BIT00           ;DIAGNOSTIC MODE
1055
1056                            ;RMDT   DRIVE TYPE REGISTER
1057
1058    100000                  NSA     =       BIT15           ;NOT SECTOR ADDRESSED=0
1059    040000                  TAP     =       BIT14           ;TAPE DRIVE = 0
1060    020000                  MOH     =       BIT13           ;MOVING HEAD = 1
1061    004000                  DRQ     =       BIT11           ;DRIVE REQUEST REQUIRED
1062
1063    020024                  SNGPRT  =       020024          ;SINGLE PORT DRIVE TYPE
1064    024024                  DULPRT  =       024024          ;DUAL PORT DRIVE TYPE
1065
1066                            ;RMOF   OFFSET REGISTER
1067
1068    010000                  FMT16   =       BIT12           ;16 BIT WORD FORMAT
1069    004000                  ECI     =       BIT11           ;ECC INHIBIT
1070    002000                  HCI     =       BIT10           ;HEADER COMPARE INHIBIT
1071    000200                  OFD     =       BIT07           ;OFFSET FORWARD
1072
1073
1074                            ;RMDC   DESIRED CYLINDER ADDRESS REGISTER
1075    001777                  CYLMSK  =       1777            ;MASK FOR CYLINDER ADDRESS
1076
1077                            ;RMMR2  MAINTENANCE REGISTER #2
1078
1079                            ;       READ ONLY BITS
1080    100000                  RQA     =       BIT15           ;PORT A REQUEST
1081    040000                  RQB     =       BIT14           ;PORT B REQUEST
1082    020000                  TAG     =       BIT13           ;TAG CONTROL
1083    010000                  TST     =       BIT12           ;COMMAND SEQUENCE TEST BIT
1084    004000                  CC      =       BIT11           ;CONTROL OR CYLINDER TAG
1085    002000                  CH      =       BIT10           ;CONTROL OR HEAD TAG
1086    001000                  BB09    =       BIT09           ;TAG BUS
1087    000400                  BB08    =       BIT08           ;TAG BUS
1088    000200                  BB07    =       BIT07           ;TAG BUS
1089    000100                  BB06    =       BIT06           ;TAG BUS
1090    000040                  BB05    =       BIT05           ;TAG BUS
1091    000020                  BB04    =       BIT04           ;TAG BUS
```

```
1092            000010          BB03    =       BIT03           ;TAG BUS
1093            000004          BB02    =       BIT02           ;TAG BUS
1094            000002          BB01    =       BIT01           ;TAG BUS
1095            000001          BB00    =       BIT00           ;TAG BUS
1096
1097
1098                            ;RMER2  ERROR REGISTER 2
1099
1100            100000          BSE     =       BIT15           ;BAD SECTOR ERROR
1101            040000          SKI     =       BIT14           ;SEEK INCOMPLETE
1102            020000          OPE     =       BIT13           ;OPERATOR PLUG ERROR
1103            010000          IVC     =       BIT12           ;INVALID COMMAND ERROR
1104            004000          LSC     =       BIT11           ;LOSS OF SYSTEM CLOCK
1105            002000          LBC     =       BIT10           ;LOSS OF BIT CLOCK
1106            000200          DVC     =       BIT07           ;DEVICE CHECK
1107            000010          DPE     =       BIT03           ;DATA PARITY ERROR
1108
1109                            .SBTTL  PROGRAM MNEMONICS
1110
1111            100000          MSE     =       BIT15           ;MANUFACTURING DETECTED SECTOR ERROR
1112            040000          USE     =       BIT14           ;USER DETECTED SECTOR ERROR
1113
1114                            .SBTTL  RM03 REGISTER INDEX VALUES
1115
1116            000000          RMCS1   =       00              ;CONTROL STATUS REGISTER
1117            000006          RMDA    =       06              ;DISK ADDRESS REGISTER
1118            000012          RMDS    =       12              ;DRIVE STATUS REGISTER
1119            000014          RMER1   =       14              ;ERROR REGISTER 1
1120            000016          RMAS    =       16              ;ATTENTION SUMMARY REGISTER
1121            000020          RMLA    =       20              ;LOOK AHEAD REGISTER
1122            000024          RMMR1   =       24              ;MAINTENANCE REGISTER
1123            000026          RMDT    =       26              ;DRIVE TYPE REGISTER
1124            000030          RMSN    =       30              ;SERIAL NUMBER REGISTER
1125            000032          RMOF    =       32              ;OFFSET REGISTER
1126            000034          RMDC    =       34              ;DESIRED CYLINDER REGISTER
1127            000036          RMCC    =       36              ;CURRENT CYLINDER REGISTER
1128            000040          RMMR2   =       40              ;MAINTENANCE REGISTER 2
1129            000042          RMER2   =       42              ;ERROR REGISTER 2
1130            000044          RMEC1   =       44              ;ECC POSITION REGISTER
1131            000046          RMEC2   =       46              ;ECC PATTERN REGISTER
1132            000050          ILRG50  =       50              ;ILLEGAL REGISTER 50
1133            000052          ILRG52  =       52              ;ILLEGAL REGISTER 52
1134            000054          ILRG54  =       54              ;ILLEGAL REGISTER 54
1135            000056          ILRG56  =       56              ;ILLEGAL REGISTER 56
1136            000060          ILRG60  =       60              ;ILLEGAL REGISTER 60
1137            000062          ILRG62  =       62              ;ILLEGAL REGISTER 62
1138            000064          ILRG64  =       64              ;ILLEGAL REGISTER 64
1139            000066          ILRG66  =       66              ;ILLEGAL REGISTER 66
1140            000070          ILRG70  =       70              ;ILLEGAL REGISTER 70
1141            000072          ILRG72  =       72              ;ILLEGAL REGISTER 72
1142            000074          ILRG74  =       74              ;ILLEGAL REGISTER 74
1143            000076          ILRG76  =       76              ;ILLEGAL REGISTER 76
1144
1145
1146            000077          IDXMSK  =       77              ;MASK FOR REGISTER INDEX NUMBER
1147
```

```
                        .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                  FSEL    =       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  037176                              $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  005324                              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

  1260  001106  000002  $PASTM: .WORD   2        ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
  1261  001110  000002  $UNITM: .WORD   2        ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
  1262  001112  000042          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
  1263        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
```

```
1320  001226  000000                $TESTN: .WORD   ATESTN  ;;TEST NUMBER
1321  001230  000000                $PASS:  .WORD   APASS   ;;PASS COUNT
1322  001232  000000                $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
1323  001234  000000                $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
1324  001236  000000                $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
1325  001240  000000                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
1326  001242                        $ETABLE:                ;;APT ENVIRONMENT TABLE
1327  001242      000               $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
1328  001243      000               $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
1329  001244  000000                $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
1330  001246  000000                $USWR:  .WORD   AUSWR   ;;USER SWITCHES
1331  001250  000000                $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
1332                                ;*                      BITS 15-11=CPU TYPE
1333                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1334                                ;*                              11/70=06,PDQ=07,Q=10
1335                                ;*                      BIT 10=REAL TIME CLOCK
1336                                ;*                      BIT  9=FLOATING POINT PROCESSOR
1337                                ;*                      BIT  8=MEMORY MANAGEMENT
1338  001252      000               $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
1339  001253      000               $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
1340                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
1341                                ;*                              900 NSEC CORE=001
1342                                ;*                              300 NSEC BIPOLAR=002
1343                                ;*                              500 NSEC MOS=003
1344  001254  000000                $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
1345                                ;*                      MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
1346  001256      000               $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
1347  001257      000               $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
1348  001260  000000                $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
1349  001262      000               $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
1350  001263      000               $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
1351  001264  000000                $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
1352  001266      000               $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
1353  001267      000               $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
1354  001270  000000                $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
1355  001272  120254                $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1356  001274  000000                $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1357  001276  176700                $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
1358  001300  000000                $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
1359  001302  000000                $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
1360  001304  000000                $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
1361  001306  000000                $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
1362  001310  000000                $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
1363  001312  000000                $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
1364  001314  000000                $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
1365  001316  000000                $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
1366  001320  000000                $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
1367  001322  000000                $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
1368  001324  000000                $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
1369  001326                        $ETEND:
1370                                .MEXIT
1371  001326  000000                CTLFG:  .WORD   0       ;
1372
1373                                ;THE REGISTER INPUT BUFFER IS USED FOR
1374                                ;STORING DRIVE STATUS
1375
```

```
1376  001330                        GETBUF:
1377
1378                                ;REGISTER INPUT BUFFER
1379  001330  000000               RMCS1I: .WORD                   ;CONTROL,STATUS REGISTER
1380  001332  000000               RMWCI:  .WORD                   ;WORD COUNT REGISTER
1381  001334  000000               RMBAI:  .WORD                   ;BUS ADDRESS REGISTER
1382  001336  000000               RMDAI:  .WORD                   ;DISK ADDRESS REGISTER
1383  001340  000000               RMCS2I: .WORD                   ;CONTROLLER STATUS REGISTER
1384  001342  000000               RMDSI:  .WORD                   ;DRIVE STATUS REGISTER
1385  001344  000000               RMER1I: .WORD                   ;ERROR REGISTER 1
1386  001346  000000               RMASI:  .WORD                   ;ATTENTION SUMMARY REGISTER
1387  001350  000000               RMLAI:  .WORD                   ;LOOK AHEAD REGISTER
1388  001352  000000               RMDBI:  .WORD                   ;DATA BUFFER
1389  001354  000000               RMMR1I: .WORD                   ;MAINTENANCE REGISTER #1
1390  001356  000000               RMDTI:  .WORD                   ;DRIVE TYPE REGISTER
1391  001360  000000               RMSNI:  .WORD                   ;SERIAL NUMBER REGISTER
1392  001362  000000               RMOFI:  .WORD                   ;OFFSET REGISTER
1393  001364  000000               RMDCI:  .WORD                   ;DESIRED CYLINDER REGISTER
1394  001366  000000               RMCCI:  .WORD                   ;CURRENT CYLINDER REGISTER
1395  001370  000000               RMMR2I: .WORD                   ;MAINTENANCE REGISTER #2
1396  001372  000000               RMER2I: .WORD                   ;ERROR REGISTER 2
1397  001374  000000               RMEC1I: .WORD                   ;ECC POSITION REGISTER
1398  001376  000000               RMEC2I: .WORD                   ;ECC PATTERN REGISTER
1399
1400                                ;THE REGISTER OUTPUT BUFFER IS USED FOR
1401                                ;ASSEMBLING DATA GOING TO REGISTER
1402
1403  001400                        PUTBUF:
1404
1405                                ;REGISTER OUTPUT BUFFER
1406  001400  000000               RMCS1O: .WORD                   ;CONTROL,STATUS REGISTER
1407  001402  000000               RMWCO:  .WORD                   ;WORD COUNT REGISTER
1408  001404  000000               RMBAO:  .WORD                   ;BUS ADDRESS REGISTER
1409  001406  000000               RMDAO:  .WORD                   ;DISK ADDRESS REGISTER
1410  001410  000000               RMCS2O: .WORD                   ;CONTROLLER STATUS REGISTER
1411  001412  000000               RMDSO:  .WORD                   ;DRIVE STATUS REGISTER
1412  001414  000000               RMER1O: .WORD                   ;ERROR REGISTER 1
1413  001416  000000               RMASO:  .WORD                   ;ATTENTION SUMMARY REGISTER
1414  001420  000000               RMLAO:  .WORD                   ;LOOK AHEAD REGISTER
1415  001422  000000               RMDBO:  .WORD                   ;DATA BUFFER
1416  001424  000000               RMMR1O: .WORD                   ;MAINTENANCE REGISTER #1
1417  001426  000000               RMDTO:  .WORD                   ;DRIVE TYPE REGISTER
1418  001430  000000               RMSNO:  .WORD                   ;SERIAL NUMBER REGISTER
1419  001432  000000               RMOFO:  .WORD                   ;OFFSET REGISTER
1420  001434  000000               RMDCO:  .WORD                   ;DESIRED CYLINDER REGISTER
1421  001436  000000               RMCCO:  .WORD                   ;CURRENT CYLINDER REGISTER
1422  001440  000000               RMMR2O: .WORD                   ;MAINTENANCE REGISTER #2
1423  001442  000000               RMER2O: .WORD                   ;ERROR REGISTER 2
1424  001444  000000               RMEC1O: .WORD                   ;ECC POSITION REGISTER
1425  001446  000000               RMEC2O: .WORD                   ;ECC PATTERN REGISTER
1426
1427                                ;EACH WORD OF THE TEST QUE CONTAINS THE DEVICE NUMBER IN
1428                                ;THE LOW BYTE AND THE ATTENTION BIT IN THE HIGH BYTE. A ZERO
1429                                ;WORD IS A BLANK AND REPRESENTS THE END OF THE QUE.
1430
1431  001450  000012               TSTQUE: .BLKW   10.             ;TEST QUE
```

```
1432
1433                                    ;MEDIA ENABLE IS SET IF THE BAD SECTOR FILES HAVE BEEN RECOVERED
1434                                    ;FOR THE UNIT UNDER TEST, OTHERWISE IT IS ZERO.
1435  001474  000000                    MEDENB: .WORD           ;MEDIA ENABLE
1436
1437                                    ;LOCATIONS "ASNDC" AND "ASNDC" CONTAIN THE CYLINDER, TRACK AND SECTOR
1438                                    ;ADDRESS ASSIGNED BY THE BAD SECTOR MODULE.
1439  001476  000000                    ASNDC:  .WORD           ;ASSIGNED DESIRED CYLINDER
1440  001500  000000                    ASNDA:  .WORD           ;ASSIGNED TRACK, AND SECTOR
1441  001502  000000                    CLKADR: .WORD           ;UNIBUS ADDRESS OF KW11 CLOCK
1442  001504  000000                    CLKVCT: .WORD           ;VECTOR ADDRESS OF KW11 CLOCK
1443
1444                                    ;THE GET INDEX TABLE CONTAINS A BYTE LIST OF REGISTERS WHICH
1445                                    ;ARE READ BY THE GET SUBROUTINE.  THE LIST IS TERMINATED BY
1446                                    ;A NEGATIVE BYTE.
1447  001506  000027                    GETINX: .BLKB   23.             ;GET INDEX TABLE
1448
1449                                    ;THE PUT INDEX TABLE ICONTAINS A BYTE LIST OF REGISTERS WHICH
1450                                    ;ARE WRITTEN BY THE PUT SUBROUTINE.  THE LIST IS TERMINATED BY
1451                                    ;A NEGATIVE BYTE.
1452  001535  000027                    PUTINX: .BLKB   23.             ;PUT INDEX TABLE
1453
1454                                    ;PUT TAGS HERE
1455
```

```
1456
1457                            .SBTTL  ERROR POINTER TABLE
1458
1459                            ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
1460                            ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
1461                            ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
1462                            ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
1463                            ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
1464
1465                            ;*      EM              ;;POINTS TO THE ERROR MESSAGE
1466                            ;*      DH              ;;POINTS TO THE DATA HEADER
1467                            ;*      DT              ;;POINTS TO THE DATA
1468                            ;*      DF              ;;POINTS TO THE DATA FORMAT
1469
1470  001564                    $ERRTB:
1471
1472
```

```
1473
1474  001564  071434                ;ERROR  1       WRONG UNIT SELECTED
1475  001566  075526                        EMT1
1476  001570  075652                        EHT1
1477  001572  075742                        EDT1
1478                                  EFT1
1479
1480                                ;ERROR  2       DEVICE WENT UNAVAILABLE
1481  001574  071440                        EMT2
1482  001576  075526                        EHT1
1483  001600  075652                        EDT1
1484  001602  075742                        EFT1
1485
1486
1487                                ;ERROR  3       DEVICE WENT NONEXISTENT
1488  001604  071446                        EMT3
1489  001606  075526                        EHT1
1490  001610  075652                        EDT1
1491  001612  075742                        EFT1
1492
1493
1494                                ;ERROR  4       CONTROLLER NOT READY
1495  001614  071454                        EMT4
1496  001616  075526                        EHT1
1497  001620  075652                        EDT1
1498  001622  075742                        EFT1
1499
1500
1501                                ;ERROR  5       DRIVE NOT READY AND GO NOT RESET
1502  001624  071462                        EMT5
1503  001626  075526                        EHT1
1504  001630  075652                        EDT1
1505  001632  075742                        EFT1
1506
1507
1508                                ;ERROR  6       UNEXPECTED VALUE FOR "ATA" STATUS
1509  001634  071470                        EMT6
1510  001636  075526                        EHT1
1511  001640  075652                        EDT1
1512  001642  075742                        EFT1
1513
1514
1515                                ;ERROR  7       BUS TIMEOUT TRYING TO READ OR WRITE REGISTER
1516  001644  071476                        EMT7
1517  001646  000000                        0
1518  001650  000000                        0
1519  001652  000000                        0
1520
1521
1522                                ;ERROR  10      DRIVE, NOT READY BUT GO IS RESET
1523  001654  071504                        EMT10
1524  001656  075526                        EHT1
1525  001660  075652                        EDT1
1526  001662  075742                        EFT1
1527
1528
```

```
1529
1530  001664  071510          ;ERROR  11      GO NOT RESET BUT DRIVE IS READY
1531  001666  075526                  EMT11
1532  001670  075652                  EHT1
1533  001672  075742                  EDT1
1534                            EFT1
1535
1536                            ;ERROR  12      INCORRECT FUNCTION CODE
1537  001674  071514                  EMT12
1538  001676  075526                  EHT1
1539  001700  075652                  EDT1
1540  001702  075742                  EFT1
1541
1542
1543                            ;ERROR  13      PARITY ERROR READING REMOTE REGISTERS
1544  001704  071522                  EMT13
1545  001706  075526                  EHT1
1546  001710  075652                  EDT1
1547  001712  075742                  EFT1
1548
1549
1550                            ;ERROR  14      TRANSFER ERROR IS INCORRECT
1551  001714  071534                  EMT14
1552  001716  075526                  EHT1
1553  001720  075652                  EDT1
1554  001722  075742                  EFT1
1555
1556
1557                            ;ERROR  15      INCORRECT WORD COUNT
1558  001724  071542                  EMT15
1559  001726  075526                  EHT1
1560  001730  075652                  EDT1
1561  001732  075742                  EFT1
1562
1563
1564                            ;ERROR  16      INGORRECT BUS ADDRESS
1565  001734  071550                  EMT16
1566  001736  075526                  EHT1
1567  001740  075652                  EDT1
1568  001742  075742                  EFT1
1569
1570
1571                            ;ERROR  17      INCORRECT LBT STATUS
1572  001744  071560                  EMT17
1573  001746  075526                  EHT1
1574  001750  075652                  EDT1
1575  001752  075742                  EFT1
1576
1577
1578                            ;ERROR  20      INCORRECT AOE
1579  001754  071570                  EMT20
1580  001756  075526                  EHT1
1581  001760  075652                  EDT1
1582  001762  075742                  EFT1
1583
1584
```

```
1585
1586  001764  071600                ;ERROR  21      INCORRECT DISK ADDRESS
1587  001766  075526                        EMT21
1588  001770  075652                        EHT1
1589  001772  075742                        EDT1
1590                                  EFT1
1591
1592                                ;ERROR  22      INCORRECT CYLINDER ADDRESS
1593  001774  071610                        EMT22
1594  001776  075526                        EHT1
1595  002000  075652                        EDT1
1596  002002  075742                        EFT1
1597
1598
1599                                ;ERROR  23      INCORRECT WLE STATUS
1600  002004  071620                        EMT23
1601  002006  075526                        EHT1
1602  002010  075652                        EDT1
1603  002012  075742                        EFT1
1604
1605
1606                                ;ERROR  24      INCORRECT UPE STATUS
1607  002014  071630                        EMT24
1608  002016  075526                        EHT1
1609  002020  075652                        EDT1
1610  002022  075742                        EFT1
1611
1612
1613                                ;ERROR  25      INCORRECT WCF STATUS
1614  002024  071640                        EMT25
1615  002026  075526                        EHT1
1616  002030  075652                        EDT1
1617  002032  075742                        EFT1
1618
1619
1620                                ;ERROR  26      INCORRECT WCE STATUS
1621  002034  071650                        EMT26
1622  002036  075526                        EHT1
1623  002040  075652                        EDT1
1624  002042  075742                        EFT1
1625
1626
1627                                ;ERROR  27      INCORRECT MDPE STATUS
1628  002044  071660                        EMT27
1629  002046  075526                        EHT1
1630  002050  075652                        EDT1
1631  002052  075742                        EFT1
1632
1633
1634                                ;ERROR  30      INCORRECT DCK STATUS
1635  002054  071670                        EMT30
1636  002056  075526                        EHT1
1637  002060  075652                        EDT1
1638  002062  075742                        EFT1
1639
1640
```

```
1641                                    ;ERROR  31      INCORRECT ECH STATUS
1642  002064  071700                              EMT31
1643  002066  075526                              EHT1
1644  002070  075652                              EDT1
1645  002072  075742                              EFT1
1646
1647
1648                                    ;ERROR  32      DLT SHOULD NOT BE SET
1649  002074  071710                              EMT32
1650  002076  075526                              EHT1
1651  002100  075652                              EDT1
1652  002102  075742                              EFT1
1653
1654
1655                                    ;ERROR  33      MXF SHOULD NOT BE SET
1656  002104  071720                              EMT33
1657  002106  075526                              EHT1
1658  002110  075652                              EDT1
1659  002112  075742                              EFT1
1660
1661
1662                                    ;ERROR  34      DTE SHOULD NOT BE SET
1663  002114  071730                              EMT34
1664  002116  075526                              EHT1
1665  002120  075652                              EDT1
1666  002122  075742                              EFT1
1667
1668
1669                                    ;ERROR  35      INCORRECT HCRC STATUS
1670  002124  071740                              EMT35
1671  002126  075526                              EHT1
1672  002130  075652                              EDT1
1673  002132  075742                              EFT1
1674
1675
1676                                    ;ERROR  36      INCORRECT HCE STATUS
1677  002134  071750                              EMT36
1678  002136  075526                              EHT1
1679  002140  075652                              EDT1
1680  002142  075742                              EFT1
1681
1682
1683                                    ;ERROR  37      INCORRECT FER STATUS
1684  002144  071760                              EMT37
1685  002146  075526                              EHT1
1686  002150  075652                              EDT1
1687  002152  075742                              EFT1
1688
1689
1690                                    ;ERROR  40      DPE SHOULD NOT BE SET (NOT A DATA COMMAND)
1691  002154  071770                              EMT40
1692  002156  075526                              EHT1
1693  002160  075652                              EDT1
1694  002162  075742                              EFT1
1695
1696
```

```
1697
1698  002164  071776                ;ERROR  41      LOST "MOL" DURING PACK ACKNOWLEDGE
1699  002166  075526                        EMT41
1700  002170  075652                        EHT1
1701  002172  075742                        EDT1
1702                                  EFT1
1703
1704                                ;ERROR  42      UNSAFE ERROR DURING PACK ACKNOWLEDGE
1705  002174  072006                        EMT42
1706  002176  075526                        EHT1
1707  002200  075652                        EDT1
1708  002202  075742                        EFT1
1709
1710
1711                                ;ERROR  43      "OPI" ERROR DURING PACK ACKNOWLEDGE
1712  002204  072020                        EMT43
1713  002206  075526                        EHT1
1714  002210  075652                        EDT1
1715  002212  075742                        EFT1
1716
1717
1718                                ;ERROR  44      "RMR" ERROR DURING PACK ACKNOWLEDGE
1719  002214  072030                        EMT44
1720  002216  075526                        EHT1
1721  002220  075652                        EDT1
1722  002222  075742                        EFT1
1723
1724
1725                                ;ERROR  45      "ILR" ERROR DURING PACK ACKNOWLEDGE
1726  002224  072040                        EMT45
1727  002226  075526                        EHT1
1728  002230  075652                        EDT1
1729  002232  075742                        EFT1
1730
1731
1732                                ;ERROR  46      "ILF" ERROR DURING PACK ACKNOWLEDGE
1733  002234  072050                        EMT46
1734  002236  075526                        EHT1
1735  002240  075652                        EDT1
1736  002242  075742                        EFT1
1737
1738
1739                                ;ERROR  47      COMPOSITE ERROR STATUS IS INCORRECT
1740  002244  072060                        EMT47
1741  002246  075526                        EHT1
1742  002250  075652                        EDT1
1743  002252  075742                        EFT1
1744
1745
1746                                ;ERROR  50      PARITY ERROR WRITING REMOTE REGISTERS
1747  002254  072066                        EMT50
1748  002256  075526                        EHT1
1749  002260  075652                        EDT1
1750  002262  075742                        EFT1
1751
1752
```

```
1753
1754  002264  072076                ;ERROR  51      INCORRECT IAE STATUS DURING SEEK COMMAND
1755  002266  075526                        EMT51
1756  002270  075652                        EHT1
1757  002272  075742                        EDT1
1758                                  EFT1
1759
1760                                ;ERROR  52      OPI ERROR DURING SEEK - MEDIUM IS NOT ON LINE
1761  002274  072110                        EMT52
1762  002276  075526                        EHT1
1763  002300  075652                        EDT1
1764  002302  075742                        EFT1
1765
1766
1767                                ;ERROR  53      OPI ERROR DURING SEEK - MEDIUM IS ON LINE, ASSUME
1768                                ;               ON CYLINDER LATCH DIDN'T RESET
1769  002304  072126                        EMT53
1770  002306  075526                        EHT1
1771  002310  075652                        EDT1
1772  002312  075742                        EFT1
1773
1774
1775                                ;ERROR  54      SEEK INCOMPLETE ERROR DURING SEEK COMMAND
1776  002314  072144                        EMT54
1777  002316  075526                        EHT1
1778  002320  075652                        EDT1
1779  002322  075742                        EFT1
1780
1781
1782                                ;ERROR  55      DEVICE CHECK DURING SEEK COMMAND
1783  002324  072154                        EMT55
1784  002326  075526                        EHT1
1785  002330  075652                        EDT1
1786  002332  075742                        EFT1
1787
1788
1789                                ;ERROR  56      PIP IS STILL SET AFTER SEEK - SKI IS RESET
1790  002334  072166                        EMT56
1791  002336  075526                        EHT1
1792  002340  075652                        EDT1
1793  002342  075742                        EFT1
1794
1795
1796                                ;ERROR  57      ATA DID NOT SET DURING SEEK COMMAND
1797  002344  072204                        EMT57
1798  002346  075526                        EHT1
1799  002350  075652                        EDT1
1800  002352  075742                        EFT1
1801
1802
1803                                ;ERROR  60      IVC ERROR DURING SEEK COMMAND - LOST
1804                                ;               VOLUME VALID
1805  002354  072214                        EMT60
1806  002356  075526                        EHT1
1807  002360  075652                        EDT1
1808  002362  075742                        EFT1
```

```
1809
1810
1811                                    ;ERROR  61      ERRONEOUS IVC ERROR DURING SEEK COMMAND -
1812                                    ;               VOLUME VALID IS STIL SET
1813  002364  072232                            EMT61
1814  002366  075526                            EHT1
1815  002370  075652                            EDT1
1816  002372  075742                            EFT1
1817
1818
1819                                    ;ERROR  62      MOL IS ZERO, BUT OPI WAS NOT
1820                                    ;               REPORTED DURING SEEK COMMAND
1821  002374  072252                            EMT62
1822  002376  075526                            EHT1
1823  002400  075652                            EDT1
1824  002402  075742                            EFT1
1825
1826
1827                                    ;ERROR  63      UNUSED
1828  002404  000000                            0
1829  002406  000000                            0
1830  002410  000000                            0
1831  002412  000000                            0
1832
1833
1834                                    ;ERROR  64      DRIVE DID NOT DETECT "IVC" ERROR DURING SEEK
1835  002414  072270                            EMT64
1836  002416  075526                            EHT1
1837  002420  075652                            EDT1
1838  002422  075742                            EFT1
1839
1840
1841                                    ;ERROR  65      DRIVE EXECUTED A SEEK WITH ERROR SET
1842  002424  072310                            EMT65
1843  002426  075526                            EHT1
1844  002430  075652                            EDT1
1845  002432  075742                            EFT1
1846
1847
1848                                    ;ERROR  66      UNEXPECTED ERROR SET IN RMER1
1849  002434  072330                            EMT66
1850  002436  075526                            EHT1
1851  002440  075652                            EDT1
1852  002442  075742                            EFT1
1853
1854
1855                                    ;ERROR  67      UNEXPECTED ERROR SET IN RMER2
1856  002444  072342                            EMT67
1857  002446  075526                            EHT1
1858  002450  075652                            EDT1
1859  002452  075742                            EFT1
1860
1861
1862                                    ;ERROR  70      ERRONEOUS "IAE" ERROR DURING RECALIBRATE
1863  002454  072354                            EMT70
1864  002456  075526                            EHT1
```

```
1865  002460  075652                    EDT1
1866  002462  075742                    EFT1
1867
1868
1869                              ;ERROR  71      "ILF" ERROR DURING RECALIBRATE
1870  002464  072364                    EMT71
1871  002466  075526                    EHT1
1872  002470  075652                    EDT1
1873  002472  075742                    EFT1
1874
1875
1876                              ;ERROR  72      "OPI" ERROR DURING RECALIBRATE DUE TO "MOL" = 0
1877  002474  072374                    EMT72
1878  002476  075526                    EHT1
1879  002500  075652                    EDT1
1880  002502  075742                    EFT1
1881
1882
1883                              ;ERROR  73      "OPI" ERROR DURING RECALIBRATE BECAUSE ON
1884                              ;               CYLINDER DIDNT DROP
1885  002504  072412                    EMT73
1886  002506  075526                    EHT1
1887  002510  075652                    EDT1
1888  002512  075742                    EFT1
1889
1890
1891                              ;ERROR  74      "IVC" ERROR DURING RECALIBRATE - "VV" = 0
1892  002514  072430                    EMT74
1893  002516  075526                    EHT1
1894  002520  075652                    EDT1
1895  002522  075742                    EFT1
1896
1897
1898                              ;ERROR  75      ERRONEOUS "IVC" ERROR DURING RECALIBRATE - "VV" = 1
1899  002524  072440                    EMT75
1900  002526  075526                    EHT1
1901  002530  075652                    EDT1
1902  002532  075742                    EFT1
1903
1904
1905                              ;ERROR  76      "SKI" ERROR DURING RECALIBRATE
1906  002534  072460                    EMT76
1907  002536  075526                    EHT1
1908  002540  075652                    EDT1
1909  002542  075742                    EFT1
1910
1911
1912                              ;ERROR  77      "DVC" OCCURRED DURING RECALIBRATE
1913  002544  072470                    EMT77
1914  002546  075526                    EHT1
1915  002550  075652                    EDT1
1916  002552  075742                    EFT1
1917
1918
1919                              ;ERROR  100     LOST "MOL" DURING RECALIBRATE - "OPI" = 0
1920  002554  072502                    EMT100
```

```
1921   002556  075526                          EHT1
1922   002560  075652                          EDT1
1923   002562  075742                          EFT1
1924
1925
1926                                   ;ERROR  101     LOST "VV" DURING RECALIBRATE - "IVC" = 0
1927   002564  072520                          EMT101
1928   002566  075526                          EHT1
1929   002570  075652                          EDT1
1930   002572  075742                          EFT1
1931
1932
1933                                   ;ERROR  102     "ATA" DID NOT SET DURING RECALIBRATE
1934   002574  072536                          EMT102
1935   002576  075526                          EHT1
1936   002600  075652                          EDT1
1937   002602  075742                          EFT1
1938
1939
1940                                   ;ERROR  103     "OM" DID NOT RESET DURING RECALIBRATE
1941   002604  072546                          EMT103
1942   002606  075526                          EHT1
1943   002610  075652                          EDT1
1944   002612  075742                          EFT1
1945
1946
1947                                   ;ERROR  104     "PIP" IS STIL SET AFTER RECALIBRATE
1948   002614  072560                          EMT104
1949   002616  075526                          EHT1
1950   002620  075652                          EDT1
1951   002622  075742                          EFT1
1952
1953
1954                                   ;ERROR  105     UNEXPECTED "ILR" ERROR DURING RECALIBRATE
1955   002624  072576                          EMT105
1956   002626  075526                          EHT1
1957   002630  075652                          EDT1
1958   002632  075742                          EFT1
1959
1960
1961                                   ;ERROR  106     UNEXPECTED "RMR" ERROR DURING RECALIBRATE
1962   002634  072606                          EMT106
1963   002636  075526                          EHT1
1964   002640  075652                          EDT1
1965   002642  075742                          EFT1
1966
1967
1968                                   ;ERROR  107     "UNS" ERROR DURING RECALIBRATE - AC POWER IS LOW
1969   002644  072616                          EMT107
1970   002646  075526                          EHT1
1971   002650  075652                          EDT1
1972   002652  075742                          EFT1
1973
1974
1975                                   ;ERROR  110     CANNOT ACCESS MASSBUS CONTROLLER VIA UNIBUS
1976   002654  072636                          EMT110
```

```
 1977  002656  075550                    EMT110
 1978  002660  075670                    EDT110
 1979  002662  075760                    EFT110
 1980
 1981
 1982                              ;ERROR  111     NONEXISTENT DEVICE
 1983  002664  072650                    EMT111
 1984  002666  075554                    EHT111
 1985  002670  075672                    EDT111
 1986  002672  075762                    EFT111
 1987
 1988
 1989                              ;ERROR  112     DEVICE NOT AVAILABLE
 1990  002674  072656                    EMT112
 1991  002676  075554                    EHT111
 1992  002700  075672                    EDT111
 1993  002702  075762                    EFT111
 1994
 1995
 1996                              ;ERROR  113     BUS TIMEOUT-NED STATUS FAILURE
 1997  002704  072664                    EMT113
 1998  002706  000000                    0
 1999  002710  000000                    0
 2000  002712  000000                    0
 2001
 2002
 2003                              ;ERROR  114     DEVICE NOT AN RM03
 2004  002714  072700                    EMT114
 2005  002716  075560                    EHT114
 2006  002720  075674                    EDT114
 2007  002722  075764                    EFT114
 2008
 2009
 2010                              ;ERROR  115     RMCS1 NOT INITIALIZED BY UNIBUS
 2011  002724  072706                    EMT115
 2012  002726  075526                    EHT1
 2013  002730  075652                    EDT1
 2014  002732  075742                    EFT1
 2015
 2016
 2017                              ;ERROR  116     RMBA NOT INITIALIZED BY UNIBUS
 2018  002734  072716                    EMT116
 2019  002736  075526                    EHT1
 2020  002740  075652                    EDT1
 2021  002742  075742                    EFT1
 2022
 2023
 2024                              ;ERROR  117     RMCS2 NOT INITIALIZED BY UNIBUS
 2025  002744  072726                    EMT117
 2026  002746  075526                    EHT1
 2027  002750  075652                    EDT1
 2028  002752  075742                    EFT1
 2029
 2030
 2031                              ;ERROR  120     RMER1 NOT INITIALIZED BY UNIBUS
 2032  002754  072736                    EMT120
```

```
2033   002756  075526                          EHT1
2034   002760  075652                          EDT1
2035   002762  075742                          EFT1
2036
2037
2038                                   ;ERROR  121     RMAS NOT INITIALIZED BY UNIBUS
2039   002764  072746                          EMT121
2040   002766  075526                          EHT1
2041   002770  075652                          EDT1
2042   002772  075742                          EFT1
2043
2044
2045                                   ;ERROR  122     RMMR1 NOT INITIALIZED BY UNIBUS
2046   002774  072756                          EMT122
2047   002776  075526                          EHT1
2048   003000  075652                          EDT1
2049   003002  075742                          EFT1
2050
2051
2052                                   ;ERROR  123     RMDS NOT INITIALIZED BY UNIBUS
2053   003004  072766                          EMT123
2054   003006  075526                          EHT1
2055   003010  075652                          EDT1
2056   003012  075742                          EFT1
2057
2058
2059                                   ;ERROR  124     RMEC2 NOT INITIALIZED BY UNIBUS
2060   003014  072776                          EMT124
2061   003016  075526                          EHT1
2062   003020  075652                          EDT1
2063   003022  075742                          EFT1
2064
2065
2066                                   ;ERROR  125     RMMR2 NOT INITIALIZED BY UNIBUS
2067   003024  073006                          EMT125
2068   003026  075526                          EHT1
2069   003030  075652                          EDT1
2070   003032  075742                          EFT1
2071
2072
2073                                   ;ERROR  126     RMCS1 NOT CLEARED BY CONTROLLER CLEAR
2074   003034  073016                          EMT126
2075   003036  075526                          EHT1
2076   003040  075652                          EDT1
2077   003042  075742                          EFT1
2078
2079
2080                                   ;ERROR  127     RMBA NOT CLEARED BY CONTROLLER CLEAR
2081   003044  073030                          EMT127
2082   003046  075526                          EHT1
2083   003050  075652                          EDT1
2084   003052  075742                          EFT1
2085
2086
2087                                   ;ERROR  130     RMCS2 NOT CLEARED BY CONTROLLER CLEAR
2088   003054  073042                          EMT130
```

```
2089   003056  075526                          EHT1
2090   003060  075652                          EDT1
2091   003062  075742                          EFT1
2092
2093
2094                                   ;ERROR  131     RMER1 NOT CLEARED BY CONTROLLER CLEAR
2095   003064  073054                          EMT131
2096   003066  075526                          EHT1
2097   003070  075652                          EDT1
2098   003072  075742                          EFT1
2099
2100
2101                                   ;ERROR  132     RMAS NOT CLEARED BY CONTROLLER CLEAR
2102   003074  073066                          EMT132
2103   003076  075526                          EHT1
2104   003100  075652                          EDT1
2105   003102  075742                          EFT1
2106
2107
2108                                   ;ERROR  133     RMMR1 NOT CLEARED BY CONTROLLER CLEAR
2109   003104  073100                          EMT133
2110   003106  075526                          EHT1
2111   003110  075652                          EDT1
2112   003112  075742                          EFT1
2113
2114
2115                                   ;ERROR  134     RMDS NOT CLEARED BY CONTROLLER CLEAR
2116   003114  073112                          EMT134
2117   003116  075526                          EHT1
2118   003120  075652                          EDT1
2119   003122  075742                          EFT1
2120
2121
2122                                   ;ERROR  135     RMEC2 NOT CLEARED BY CONTROLLER CLEAR
2123   003124  073124                          EMT135
2124   003126  075526                          EHT1
2125   003130  075652                          EDT1
2126   003132  075742                          EFT1
2127
2128
2129                                   ;ERROR  136     RMMR2 NOT CLEARED BY CONTROLLER CLEAR
2130   003134  073136                          EMT136
2131   003136  075526                          EHT1
2132   003140  075652                          EDT1
2133   003142  075742                          EFT1
2134
2135
2136                                   ;ERROR  137     RMCS1 NOT CLEARED BY ERROR CLEAR
2137   003144  073150                          EMT137
2138   003146  075526                          EHT1
2139   003150  075652                          EDT1
2140   003152  075742                          EFT1
2141
2142
2143                                   ;ERROR  140     RMCS2 NOT CLEARED BY ERROR CLEAR
2144   003154  073160                          EMT140
```

```
2145  003156  075526                    EHT1
2146  003160  075652                    EDT1
2147  003162  075742                    EFT1
2148
2149
2150                          ;ERROR  141     RMCS1 NOT CLEARED BY DRIVE CLEAR
2151  003164  073170                    EMT141
2152  003166  075526                    EHT1
2153  003170  075652                    EDT1
2154  003172  075742                    EFT1
2155
2156
2157                          ;ERROR  142     RMDS NOT CLEARED BY DRIVE CLEAR
2158  003174  073200                    EMT142
2159  003176  075526                    EHT1
2160  003200  075652                    EDT1
2161  003202  075742                    EFT1
2162
2163
2164                          ;ERROR  143     RMER1 NOT CLEARED BY DRIVE CLEAR
2165  003204  073210                    EMT143
2166  003206  075526                    EHT1
2167  003210  075652                    EDT1
2168  003212  075742                    EFT1
2169
2170
2171                          ;ERROR  144     RMAS NOT CLEARED BY DRIVE CLEAR
2172  003214  073220                    EMT144
2173  003216  075526                    EHT1
2174  003220  075652                    EDT1
2175  003222  075742                    EFT1
2176
2177
2178                          ;ERROR  145     RMMR1  NOT CLEARED BY DRIVE CLEAR
2179  003224  073230                    EMT145
2180  003226  075526                    EHT1
2181  003230  075652                    EDT1
2182  003232  075742                    EFT1
2183
2184
2185                          ;ERROR  146     RMMR2 NOT CLEARED BY DRIVE CLEAR
2186  003234  073240                    EMT146
2187  003236  075526                    EHT1
2188  003240  075652                    EDT1
2189  003242  075742                    EFT1
2190
2191
2192                          ;ERROR  147     RMER2 NOT CLEARED BY DRIVE CLEAR
2193  003244  073250                    EMT147
2194  003246  075526                    EHT1
2195  003250  075652                    EDT1
2196  003252  075742                    EFT1
2197
2198
2199                          ;ERROR  150     RMEC2 NOT CLEARED BY DRIVE CLEAR
2200  003254  073260                    EMT150
```

```
2201  003256  075526                    EHT1
2202  003260  075652                    EDT1
2203  003262  075742                    EFT1
2204
2205
2206                              ;ERROR  151     MEDIUM NOT ON LINE
2207  003264  073270                    EMT151
2208  003266  075526                    EHT1
2209  003270  075652                    EDT1
2210  003272  075742                    EFT1
2211
2212
2213                              ;ERROR  152     DRIVE FAULT
2214  003274  073302                    EMT152
2215  003276  075526                    EHT1
2216  003300  075652                    EDT1
2217  003302  075742                    EFT1
2218
2219
2220                              ;ERROR  153     UNSAFE SHOULD BE SET BECAUSE DVC IS SET
2221  003304  073314                    EMT153
2222  003306  075526                    EHT1
2223  003310  075652                    EDT1
2224  003312  075742                    EFT1
2225
2226
2227                              ;ERROR  154     UNSAFE SHOULD NOT BE SET, AC IS LOW
2228  003314  073332                    EMT154
2229  003316  075526                    EHT1
2230  003320  075652                    EDT1
2231  003322  075742                    EFT1
2232
2233
2234                              ;ERROR  155     VOLUME VALID NOT SET BY PACK ACK
2235  003324  073350                    EMT155
2236  003326  075526                    EHT1
2237  003330  075652                    EDT1
2238  003332  075742                    EFT1
2239
2240
2241                              ;ERROR  156     OFFSET MODE NOT SET BY OFFSET COMMAND
2242  003334  073362                    EMT156
2243  003336  075526                    EHT1
2244  003340  075652                    EDT1
2245  003342  075742                    EFT1
2246
2247
2248                              ;ERROR  157     OFFSET MODE NOT RESET BY RTC COMMAND
2249  003344  073374                    EMT157
2250  003346  075526                    EHT1
2251  003350  075652                    EDT1
2252  003352  075742                    EFT1
2253
2254
2255                              ;ERROR  160     RMOF NOT RESET BY RIP COMMAND
2256  003354  073406                    EMT160
```

```
2257   003356   075526                  EHT1
2258   003360   075652                  EDT1
2259   003362   075742                  EFT1
2260
2261
2262                            ;ERROR  161     RMDA NOT RESET BY RIP COMMAND
2263   003364   073416                  EMT161
2264   003366   075526                  EHT1
2265   003370   075652                  EDT1
2266   003372   075742                  EFT1
2267
2268
2269                            ;ERROR  162     RMDC NOT RESET BY RIP COMMAND
2270   003374   073430                  EMT162
2271   003376   075526                  EHT1
2272   003400   075652                  EDT1
2273   003402   075742                  EFT1
2274
2275
2276                            ;ERROR 163      DATA WAS ECC CORRECTED BUT DOES NOT COMPARE WITH
2277                            ;               WRITE BUFFER
2278   003404   075322                  EMT336
2279   003406   075610                  EHT336
2280   003410   075706                  EDT336
2281   003412   075776                  EFT336
2282
2283
2284                            ;ERROR  164     OPI SHOULD NOT BE SET
2285   003414   073452                  EMT164
2286   003416   075526                  EHT1
2287   003420   075652                  EDT1
2288   003422   075742                  EFT1
2289
2290
2291                            ;ERROR  165     IVC SHOULD NOT BE SET
2292   003424   073460                  EMT165
2293   003426   075526                  EHT1
2294   003430   075652                  EDT1
2295   003432   075742                  EFT1
2296
2297
2298                            ;ERROR  166     IAE SHOULD NOT BE SET
2299   003434   073466                  EMT166
2300   003436   075526                  EHT1
2301   003440   075652                  EDT1
2302   003442   075742                  EFT1
2303
2304
2305                            ;ERROR  167     NEM SHOULD NOT BE SET
2306   003444   073474                  EMT167
2307   003446   075526                  EHT1
2308   003450   075652                  EDT1
2309   003452   075742                  EFT1
2310
2311
2312                            ;ERROR 170      UNUSED
```

```
2313  003454  000000                    0
2314  003456  000000                    0
2315  003460  000000                    0
2316  003462  000000                    0
2317
2318
2319                          ;ERROR  171     "ATA" NOT SET DURING RETURN TO CENTERLINE
2320  003464  073512                    EMT171
2321  003466  075526                    EHT1
2322  003470  075652                    EDT1
2323  003472  075742                    EFT1
2324
2325
2326                          ;ERROR  172     "ATA" NOT SET BY OFFSET COMMAND
2327  003474  073522                    EMT172
2328  003476  075526                    EHT1
2329  003500  075652                    EDT1
2330  003502  075742                    EFT1
2331
2332
2333                          ;ERROR  173     RMER2 NOT INITIALIZED BY UNIBUS INIT
2334  003504  073532                    EMT173
2335  003506  075526                    EHT1
2336  003510  075652                    EDT1
2337  003512  075742                    EFT1
2338
2339
2340                          ;ERROR  174     RMER2 NOT INITIALIZED BY CONTROLLER CLEAR
2341  003514  073542                    EMT174
2342  003516  075526                    EHT1
2343  003520  075652                    EDT1
2344  003522  075742                    EFT1
2345
2346
2347                          ;ERROR  175     SELECTED DEVICE IS IN WRITE PROTECT
2348  003524  073554                    EMT175
2349  003526  075526                    EHT1
2350  003530  075652                    EDT1
2351  003532  075742                    EFT1
2352
2353
2354                          ;ERROR  176     CANNOT SET DIAGNOSTIC MODE
2355  003534  073562                    EMT176
2356  003536  075526                    EHT1
2357  003540  075652                    EDT1
2358  003542  075742                    EFT1
2359
2360
2361                          ;ERROR  177     INCORRECT "MOL" STATUS DURING DIAGNOSTIC MODE
2362  003544  073570                    EMT177
2363  003546  075526                    EHT1
2364  003550  075652                    EDT1
2365  003552  075742                    EFT1
2366
2367
2368                          ;ERROR  200     INCORRECT "PIP" STATUS DURING DIAGNOSTIC MODE
```

```
2369  003554  073602                        EMT200
2370  003556  075526                        EHT1
2371  003560  075652                        EDT1
2372  003562  075742                        EFT1
2373
2374
2375                              ;ERROR  201     INCORRECT "WRL" STATUS DURING DIAGNOSTIC MODE
2376  003564  073614                        EMT201
2377  003566  075526                        EHT1
2378  003570  075652                        EDT1
2379  003572  075742                        EFT1
2380
2381
2382                              ;ERROR  202     INCORRECT "SKI" STATUS DURING DIAGNOSTIC MODE
2383  003574  073626                        EMT202
2384  003576  075526                        EHT1
2385  003600  075652                        EDT1
2386  003602  075742                        EFT1
2387
2388
2389                              ;ERROR  203     INCORRECT "DVC" STATUS DURING DIAGNOSTIC MODE
2390  003604  073640                        EMT203
2391  003606  075526                        EHT1
2392  003610  075652                        EDT1
2393  003612  075742                        EFT1
2394
2395
2396                              ;ERROR  204     "VV" WAS NOT RESET BY MAINTENANCE UNIT READY
2397  003614  073652                        EMT204
2398  003616  075526                        EHT1
2399  003620  075652                        EDT1
2400  003622  075742                        EFT1
2401
2402
2403                              ;ERROR  205     SELECTED DEVICE HAS A PERSISTENT "SKI" ERROR
2404  003624  073670                        EMT205
2405  003626  075526                        EHT1
2406  003630  075652                        EDT1
2407  003632  075742                        EFT1
2408
2409
2410                              ;ERROR  206     "LBC" DID NOT SET DURING DIAGNOSTIC MODE
2411  003634  073700                        EMT206
2412  003636  075526                        EHT1
2413  003640  075652                        EDT1
2414  003642  075742                        EFT1
2415
2416
2417                              ;ERROR  207     UNEXPECTED LOSS OF "MOL" - MEDIUM IS OFF LINE
2418  003644  073710                        EMT207
2419  003646  075526                        EHT1
2420  003650  075652                        EDT1
2421  003652  075742                        EFT1
2422
2423
2424                              ;ERROR  210     UNEXPECTED LOSS OF VOLUME VALID - "VV" = 0
```

```
2425  003654  073722                    EMT210
2426  003656  075526                    EHT1
2427  003660  075652                    EDT1
2428  003662  075742                    EFT1
2429
2430
2431                          ;ERROR  211     UNEXPECTED MECHANICAL MOTION - "PIP" = 1
2432  003664  073730                    EMT211
2433  003666  075526                    EHT1
2434  003670  075652                    EDT1
2435  003672  075742                    EFT1
2436
2437
2438                          ;ERROR  212     UNEXPECTED DEVICE FAULT - "DVC" = 1
2439  003674  073744                    EMT212
2440  003676  075526                    EHT1
2441  003700  075652                    EDT1
2442  003702  075742                    EFT1
2443
2444
2445                          ;ERROR  213     UNEXPECTED SEEK INCOMPLETE ERROR - "SKI" = 1
2446  003704  073760                    EMT213
2447  003706  075526                    EHT1
2448  003710  075652                    EDT1
2449  003712  075742                    EFT1
2450
2451
2452                          ;ERROR  214     DRIVE EXECUTED A RECALIBRATE WITH ERROR SET
2453  003714  073770                    EMT214
2454  003716  075540                    EHT2
2455  003720  075662                    EDT2
2456  003722  075752                    EFT2
2457
2458
2459                          ;ERROR  215     DRIVE DID NOT DETECT "IVC" ERROR DURING RECALIBRATE
2460  003724  074010                    EMT215
2461  003726  075540                    EHT2
2462  003730  075662                    EDT2
2463  003732  075752                    EFT2
2464
2465
2466                          ;ERROR  216     INCORRECT "IVC" STATUS
2467  003734  074022                    EMT216
2468  003736  075526                    EHT1
2469  003740  075652                    EDT1
2470  003742  075742                    EFT1
2471
2472
2473                          ;ERROR  217     INCORRECT "IAE" STATUS
2474  003744  074032                    EMT217
2475  003746  075526                    EHT1
2476  003750  075652                    EDT1
2477  003752  075742                    EFT1
2478
2479
2480                          ;ERROR 220      INCORRECT "WLE" STATUS
```

```
2481   003754   074042                          EMT220
2482   003756   075526                          EHT1
2483   003760   075652                          EDT1
2484   003762   075742                          EFT1
2485
2486
2487                                    ;ERROR  221     INCORRECT "OPI" STATUS
2488   003764   074052                          EMT221
2489   003766   075526                          EHT1
2490   003770   075652                          EDT1
2491   003772   075742                          EFT1
2492
2493
2494                                    ;ERROR  222     RM03 DID NOT DETECT RMR ERROR
2495   003774   074062                          EMT222
2496   003776   075526                          EHT1
2497   004000   075652                          EDT1
2498   004002   075742                          EFT1
2499
2500
2501                                    ;ERROR  223     RM03 DID NOT DETECT PARITY ERROR ON MASSBUS CONTROL BUS
2502   004004   074072                          EMT223
2503   004006   075564                          EHT223
2504   004010   075676                          EDT223
2505   004012   075766                          EFT223
2506
2507
2508                                    ;ERROR  224     UNUSED
2509   004014   000000                          0
2510   004016   000000                          0
2511   004020   000000                          0
2512   004022   000000                          0
2513
2514
2515                                    ;ERROR  225     UNUSED
2516   004024   000000                          0
2517   004026   000000                          0
2518   004030   000000                          0
2519   004032   000000                          0
2520
2521
2522                                    ;ERROR  226     UNUSED
2523   004034   000000                          0
2524   004036   000000                          0
2525   004040   000000                          0
2526   004042   000000                          0
2527
2528
2529                                    ;ERROR  227     UNUSED
2530   004044   000000                          0
2531   004046   000000                          0
2532   004050   000000                          0
2533   004052   000000                          0
2534
2535
2536                                    ;ERROR  230     UNUSED
```

```
2537   004054   000000                          0
2538   004056   000000                          0
2539   004060   000000                          0
2540   004062   000000                          0
2541
2542
2543                                  ;ERROR  231     UNUSED
2544   004064   000000                          0
2545   004066   000000                          0
2546   004070   000000                          0
2547   004072   000000                          0
2548
2549
2550                                  ;ERROR  232     UNUSED
2551   004074   000000                          0
2552   004076   000000                          0
2553   004100   000000                          0
2554   004102   000000                          0
2555
2556
2557                                  ;ERROR  233     UNUSED
2558   004104   000000                          0
2559   004106   000000                          0
2560   004110   000000                          0
2561   004112   000000                          0
2562
2563
2564                                  ;ERROR  234     UNUSED
2565   004114   000000                          0
2566   004116   000000                          0
2567   004120   000000                          0
2568   004122   000000                          0
2569
2570
2571                                  ;ERROR  235     UNUSED
2572   004124   000000                          0
2573   004126   000000                          0
2574   004130   000000                          0
2575   004132   000000                          0
2576
2577
2578                                  ;ERROR  236     UNUSED
2579   004134   000000                          0
2580   004136   000000                          0
2581   004140   000000                          0
2582   004142   000000                          0
2583
2584
2585                                  ;ERROR  237     UNUSED
2586   004144   000000                          0
2587   004146   000000                          0
2588   004150   000000                          0
2589   004152   000000                          0
2590
2591
2592                                  ;ERROR  240     UNUSED
```

```
2593  004154  000000                    0
2594  004156  000000                    0
2595  004160  000000                    0
2596  004162  000000                    0
2597
2598
2599                              ;ERROR  241     UNUSED
2600  004164  000000                    0
2601  004166  000000                    0
2602  004170  000000                    0
2603  004172  000000                    0
2604
2605
2606                              ;ERROR  242     UNUSED
2607  004174  000000                    0
2608  004176  000000                    0
2609  004200  000000                    0
2610  004202  000000                    0
2611
2612
2613                              ;ERROR  243     UNUSED
2614  004204  000000                    0
2615  004206  000000                    0
2616  004210  000000                    0
2617  004212  000000                    0
2618
2619
2620                              ;ERROR  244     UNUSED
2621  004214  000000                    0
2622  004216  000000                    0
2623  004220  000000                    0
2624  004222  000000                    0
2625
2626
2627                              ;ERROR  245     UNUSED
2628  004224  000000                    0
2629  004226  000000                    0
2630  004230  000000                    0
2631  004232  000000                    0
2632
2633
2634                              ;ERROR  246     "ATA" NOT RESET BY GO WHEN "ERR" = 0
2635  004234  074146                    EMT246
2636  004236  075526                    EHT1
2637  004240  075652                    EDT1
2638  004242  075742                    EFT1
2639
2640
2641                              ;ERROR  247     "ATA" NOT RESET BY WRITING RMAS
2642  004244  074156                    EMT247
2643  004246  075526                    EHT1
2644  004250  075652                    EDT1
2645  004252  075742                    EFT1
2646
2647
2648                              ;ERROR  250     "ATA" WAS RESET BY GO WHEN "ERR" = 1
```

```
2649   004254   074170                          EMT250
2650   004256   075526                          EHT1
2651   004260   075652                          EDT1
2652   004262   075742                          EFT1
2653
2654
2655                                    ;ERROR  251     PROGRAM INTERRUPT WAS NOT GENERATED
2656   004264   074204                          EMT251
2657   004266   075540                          EHT2
2658   004270   075662                          EDT2
2659   004272   075752                          EFT2
2660
2661
2662                                    ;ERROR  252     PROGRAM INTERRUPT SHOULD NOT HAVE BEEN GENERATED
2663   004274   074212                          EMT252
2664   004276   075540                          EHT2
2665   004300   075662                          EDT2
2666   004302   075752                          EFT2
2667
2668
2669                                    ;ERROR  253     OFFSET MODE WAS NOT RESET BY WRITING RMDC
2670   004304   074220                          EMT253
2671   004306   075526                          EHT1
2672   004310   075652                          EDT1
2673   004312   075742                          EFT1
2674
2675
2676                                    ;ERROR  254     INCORRECT "ILF" STATUS
2677   004314   074236                          EMT254
2678   004316   075526                          EHT1
2679   004320   075652                          EDT1
2680   004322   075742                          EFT1
2681
2682
2683                                    ;ERROR  255     INCORRECT "ATA" STATUS
2684   004324   074246                          EMT255
2685   004326   075526                          EHT1
2686   004330   075652                          EDT1
2687   004332   075742                          EFT1
2688
2689
2690                                    ;ERROR  256     INCORRECT "ILR" STATUS
2691   004334   074256                          EMT256
2692   004336   075576                          EHT256
2693   004340   075676                          EDT223
2694   004342   075766                          EFT223
2695
2696
2697                                    ;ERROR  257     INVALID IAE STATUS DURING SEARCH COMMAND
2698   004344   074266                          EMT257
2699   004346   075526                          EHT1
2700   004350   075652                          EDT1
2701   004352   075742                          EFT1
2702
2703
2704                                    ;ERROR  260     "IVC" WAS NOT DETECTED DURING SEARCH COMMAND
```

```
2705  004354  074300                    EMT260
2706  004356  075526                    EHT1
2707  004360  075652                    EDT1
2708  004362  075742                    EFT1
2709
2710
2711                          ;ERROR  261     DRIVE EXECUTED SEARCH WITH ERROR SET
2712  004364  074312                    EMT261
2713  004366  075526                    EHT1
2714  004370  075652                    EDT1
2715  004372  075742                    EFT1
2716
2717
2718                          ;ERROR  262     "LBC" ERROR NOT SET DURING DIAGNOSTIC MODE
2719  004374  074332                    EMT262
2720  004376  075526                    EHT1
2721  004400  075652                    EDT1
2722  004402  075742                    EFT1
2723
2724
2725                          ;ERROR  263     "SKI" ERROR DURING SEARCH COMMAND
2726  004404  074342                    EMT263
2727  004406  075526                    EHT1
2728  004410  075652                    EDT1
2729  004412  075742                    EFT1
2730
2731
2732                          ;ERROR  264     "IVC" ERROR DURING SEARCH - LOST VOLUME VALID
2733  004414  074352                    EMT264
2734  004416  075526                    EHT1
2735  004420  075652                    EDT1
2736  004422  075742                    EFT1
2737
2738
2739                          ;ERROR  265     ERRONEOUS IVC ERROR DURING SEARCH - VOLUME IS VALID
2740  004424  074372                    EMT265
2741  004426  075526                    EHT1
2742  004430  075652                    EDT1
2743  004432  075742                    EFT1
2744
2745
2746                          ;ERROR  266     DEVICE FAULT (DVC) DURING SEARCH
2747  004434  074412                    EMT266
2748  004436  075526                    EHT1
2749  004440  075652                    EDT1
2750  004442  075742                    EFT1
2751
2752
2753                          ;ERROR  267     SKI SHOULD HAVE BEEN SET BECAUSE CYLINDER
2754                          ;               ADDRESS IS TOO LARGE
2755  004444  074424                    EMT267
2756  004446  075526                    EHT1
2757  004450  075652                    EDT1
2758  004452  075742                    EFT1
2759
2760
```

```
2761                                    ;ERROR  270     OPI ERROR DURING SEARCH BECAUSE MOL = 0
2762   004454  074442                           EMT270
2763   004456  075526                           EHT1
2764   004460  075652                           EDT1
2765   004462  075742                           EFT1
2766
2767
2768                                    ;ERROR  271     OPI ERROR DURING SEARCH BECAUSE ON CYLINDER
2769                                    ;               DIDN'T DROP
2770   004464  074456                           EMT271
2771   004466  075526                           EHT1
2772   004470  075652                           EDT1
2773   004472  075742                           EFT1
2774
2775
2776                                    ;ERROR  272     LOST MOL DURING SEARCH, OPI IS NOT SET
2777   004474  074474                           EMT272
2778   004476  075526                           EHT1
2779   004500  075652                           EDT1
2780   004502  075742                           EFT1
2781
2782
2783                                    ;ERROR  273     PIP STIL SET AFTER SEARCH
2784   004504  074512                           EMT273
2785   004506  075526                           EHT1
2786   004510  075652                           EDT1
2787   004512  075742                           EFT1
2788
2789
2790                                    ;ERROR  274     PARITY ERROR OCCURRED WHILE WRITING REMOTE
2791                                    ;               REGISTERS BUT MXF DID NOT SET
2792   004514  074530                           EMT274
2793   004516  075526                           EHT1
2794   004520  075652                           EDT1
2795   004522  075742                           EFT1
2796
2797
2798                                    ;ERROR  275     MXF ERROR - COMPOSITE ERROR OCCURRED BEFORE DATA
2799                                    ;               COMMAND STARTED
2800   004524  074546                           EMT275
2801   004526  075526                           EHT1
2802   004530  075652                           EDT1
2803   004532  075742                           EFT1
2804
2805
2806                                    ;ERROR  276     "OPI" ERROR DURING DATA TRANSFER BECAUSE "MOL" WAS
2807                                    ;               ZERO
2808   004534  074560                           EMT276
2809   004536  075526                           EHT1
2810   004540  075652                           EDT1
2811   004542  075742                           EFT1
2812
2813
2814                                    ;ERROR  277     "OPI" ERROR DURING DATA TRANSFER BECAUSE 1) ON
2815                                    ;               CYLINDER DIDN'T DROP OR 2) SEARCH TIMED OUT OR
2816                                    ;               3) RUN TIMED OUT
```

```
2817  004544  074574                    EMT277
2818  004546  075526                    EHT1
2819  004550  075652                    EDT1
2820  004552  075742                    EFT1
2821
2822
2823                              ;ERROR  300     "IVC" ERROR DURING DATA TRANSFER BECAUSE VOLUME
2824                              ;               WAS NOT VALID
2825  004554  074612                    EMT300
2826  004556  075526                    EHT1
2827  004560  075652                    EDT1
2828  004562  075742                    EFT1
2829
2830
2831                              ;ERROR  301     ERRONEOUS "IVC" ERROR DURING DATA TRANSFER - VOLUME
2832                              ;               IS VALID
2833  004564  074632                    EMT301
2834  004566  075526                    EHT1
2835  004570  075652                    EDT1
2836  004572  075742                    EFT1
2837
2838
2839                              ;ERROR  302     "ILR" ERROR DURING DATA TRANSFER
2840  004574  074654                    EMT302
2841  004576  075526                    EHT1
2842  004600  075652                    EDT1
2843  004602  075742                    EFT1
2844
2845
2846                              ;ERROR  303     "ILF" ERROR DURING DATA TRANSFER
2847  004604  074666                    EMT303
2848  004606  075526                    EHT1
2849  004610  075652                    EDT1
2850  004612  075742                    EFT1
2851
2852
2853                              ;ERROR  304     "RMR" ERROR DURING DATA TRANSFER
2854  004614  074700                    EMT304
2855  004616  075526                    EHT1
2856  004620  075652                    EDT1
2857  004622  075742                    EFT1
2858
2859
2860                              ;ERROR  305     INCORRECT "IAE" STATUS DURING DATA TRANSFER
2861  004624  074712                    EMT305
2862  004626  075526                    EHT1
2863  004630  075652                    EDT1
2864  004632  075742                    EFT1
2865
2866
2867                              ;ERROR  306     "SKI" ERROR DURING DATA TRANSFER
2868  004634  074724                    EMT306
2869  004636  075526                    EHT1
2870  004640  075652                    EDT1
2871  004642  075742                    EFT1
2872
```

```
2873
2874                                    ;ERROR  307     DRIVE DID NOT DETECT SKI ERROR DUE TO CYLINDER
2875  004644  074734                              EMT307
2876  004646  075526                              EHT1
2877  004650  075652                              EDT1
2878  004652  075742                              EFT1
2879
2880
2881                                    ;ERROR  310     DEVICE FAULT DURING DATA TRANSFER
2882  004654  074754                              EMT310
2883  004656  075526                              EHT1
2884  004660  075652                              EDT1
2885  004662  075742                              EFT1
2886
2887
2888                                    ;ERROR  311     LOSS OF BIT CLOCK DURING DATA TRANSFER
2889  004664  074766                              EMT311
2890  004666  075526                              EHT1
2891  004670  075652                              EDT1
2892  004672  075742                              EFT1
2893
2894
2895                                    ;ERROR  312     LOSS OF SYSTEM CLOCK DURING DATA TRANSFER
2896  004674  075000                              EMT312
2897  004676  075526                              EHT1
2898  004700  075652                              EDT1
2899  004702  075742                              EFT1
2900
2901
2902                                    ;ERROR  313     UNSAFE ERROR DURING DATA TRANSFER (DVC = 0)
2903  004704  075012                              EMT313
2904  004706  075526                              EHT1
2905  004710  075652                              EDT1
2906  004712  075742                              EFT1
2907
2908
2909                                    ;ERROR  314     DRIVE TIMING ERROR DURING DATA TRANSFER
2910  004714  075032                              EMT314
2911  004716  075526                              EHT1
2912  004720  075652                              EDT1
2913  004722  075742                              EFT1
2914
2915
2916                                    ;ERROR  315     WRITE LOCK ERROR
2917  004724  075044                              EMT315
2918  004726  075526                              EHT1
2919  004730  075652                              EDT1
2920  004732  075742                              EFT1
2921
2922
2923                                    ;ERROR  316     ERRONEOUS WRITE LOCK ERROR
2924  004734  075056                              EMT316
2925  004736  075526                              EHT1
2926  004740  075652                              EDT1
2927  004742  075742                              EFT1
2928
```

```
2929
2930                                    ;ERROR  317     HEADER CRC ERROR DURING DATA TRANSFER
2931   004744  075070                           EMT317
2932   004746  075526                           EHT1
2933   004750  075652                           EDT1
2934   004752  075742                           EFT1
2935
2936
2937                                    ;ERROR  320     FORMAT ERROR DURING DATA TRANSFER
2938   004754  075100                           EMT320
2939   004756  075526                           EHT1
2940   004760  075652                           EDT1
2941   004762  075742                           EFT1
2942
2943
2944                                    ;ERROR  321     HEADER COMPARE ERROR DURING DATA TRANSFER
2945   004764  075110                           EMT321
2946   004766  075526                           EHT1
2947   004770  075652                           EDT1
2948   004772  075742                           EFT1
2949
2950
2951                                    ;ERROR  322     HEADER ERRORS SHOULD NOT BE SET
2952   004774  075120                           EMT322
2953   004776  075526                           EHT1
2954   005000  075652                           EDT1
2955   005002  075742                           EFT1
2956
2957
2958                                    ;ERROR  323     DATA CHECK ERROR DURING DATA TRANSFER
2959   005004  075126                           EMT323
2960   005006  075526                           EHT1
2961   005010  075652                           EDT1
2962   005012  075742                           EFT1
2963
2964
2965                                    ;ERROR  324     CORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
2966   005014  075136                           EMT324
2967   005016  075526                           EHT1
2968   005020  075652                           EDT1
2969   005022  075742                           EFT1
2970
2971
2972                                    ;ERROR  325     UNCORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
2973   005024  075150                           EMT325
2974   005026  075526                           EHT1
2975   005030  075652                           EDT1
2976   005032  075742                           EFT1
2977
2978
2979                                    ;ERROR  326     DATA PARITY ERROR DURING READ COMMMAND
2980   005034  075162                           EMT326
2981   005036  075526                           EHT1
2982   005040  075652                           EDT1
2983   005042  075742                           EFT1
2984
```

```
2985
2986                                    ;ERROR  327     OFFSET MODE NOT RESET BY WRITE COMMAND
2987   005044  075200                             EMT327
2988   005046  075526                             EHT1
2989   005050  075652                             EDT1
2990   005052  075742                             EFT1
2991
2992
2993                                    ;ERROR  330     DATA PARITY ERROR DURING WRITE COMMAND
2994   005054  075212                             EMT330
2995   005056  075526                             EHT1
2996   005060  075652                             EDT1
2997   005062  075742                             EFT1
2998
2999
3000                                    ;ERROR  331     WRITE CLOCK FALURE DURING WRITE COMMAND
3001   005064  075222                             EMT331
3002   005066  075526                             EHT1
3003   005070  075652                             EDT1
3004   005072  075742                             EFT1
3005
3006
3007                                    ;ERROR  332     DATA LATE ERROR DURING DATA TRANSFER
3008   005074  075234                             EMT332
3009   005076  075526                             EHT1
3010   005100  075652                             EDT1
3011   005102  075742                             EFT1
3012
3013
3014                                    ;ERROR  333     PIP STIL SET AFTER DATA TRANSFER - SKI = 0
3015   005104  075246                             EMT333
3016   005106  075526                             EHT1
3017   005110  075652                             EDT1
3018   005112  075742                             EFT1
3019
3020
3021                                    ;ERROR  334     LOST MOL DURING DATA TRANSFER - OPI = 0
3022   005114  075264                             EMT334
3023   005116  075526                             EHT1
3024   005120  075652                             EDT1
3025   005122  075742                             EFT1
3026
3027
3028                                    ;ERROR  335     LOST VOLUME VALID DURING DATA TRANSFER - IVC  = 0
3029   005124  075302                             EMT335
3030   005126  075526                             EHT1
3031   005130  075652                             EDT1
3032   005132  075742                             EFT1
3033
3034
3035                                    ;ERROR  336     DATA READ DOES NOT COMPARE WITH DATA WRITTEN
3036   005134  075322                             EMT336
3037   005136  075610                             EHT336
3038   005140  075706                             EDT336
3039   005142  075776                             EFT336
3040
```

```
3041
3042                                    ;ERROR  337     WRITE CHECK ERROR NOT DETECTED
3043  005144  075332                              EMT337
3044  005146  075622                              EHT337
3045  005150  075716                              EDT337
3046  005152  076006                              EFT337
3047
3048
3049                                    ;ERROR  340     WRITE CHECK ERROR AT UNEXPECTED ADDRESS
3050  005154  075342                              EMT340
3051  005156  075610                              EHT336
3052  005160  075706                              EDT336
3053  005162  075776                              EFT336
3054
3055
3056                                    ;ERROR  341     INCORRECT DATA DURING WRITE CHECK ERROR
3057  005164  075354                              EMT341
3058  005166  075610                              EHT336
3059  005170  075706                              EDT336
3060  005172  075776                              EFT336
3061
3062
3063                                    ;ERROR  342     "IVC" ERROR NOT DETECTED DURING DATA TRANSFER
3064  005174  075362                              EMT342
3065  005176  075526                              EHT1
3066  005200  075652                              EDT1
3067  005202  075742                              EFT1
3068
3069
3070                                    ;ERROR  343     "FER" NOT DETECTED DURING DATA TRANSFER
3071  005204  075374                              EMT343
3072  005206  075526                              EHT1
3073  005210  075652                              EDT1
3074  005212  075742                              EFT1
3075
3076
3077                                    ;ERROR  344     "HCE" NOT DETECTED DURING DATA TRANSFER
3078  005214  075406                              EMT344
3079  005216  075634                              EHT344
3080  005220  075726                              EDT344
3081  005222  076016                              EFT344
3082
3083
3084                                    ;ERROR  345     "BSE" NOT DETECTED DURING DATA TRANSFER
3085  005224  075420                              EMT345
3086  005226  075526                              EHT1
3087  005230  075652                              EDT1
3088  005232  075742                              EFT1
3089
3090
3091                                    ;ERROR  346     HEADER ERROR WAS DETECTED W/ HCI SET
3092  005234  075430                              EMT346
3093  005236  075526                              EHT1
3094  005240  075652                              EDT1
3095  005242  075742                              EFT1
3096
```

```
3097
3098                                    ;ERROR  347     DATA TRANSFER NOT ABORTED W/ COMP ERROR SET
3099   005244  075444                             EMT347
3100   005246  075526                             EHT1
3101   005250  075652                             EDT1
3102   005252  075742                             EFT1
3103
3104
3105                                    ;ERROR  350     LOST VOLUME VALID DURING SEARCH - "IVC" = 0
3106   005254  075456                             EMT350
3107   005256  075526                             EHT1
3108   005260  075652                             EDT1
3109   005262  075742                             EFT1
3110
3111
3112                                    ;ERROR  351     "ATA" DID NOT SET DURING SEARCH
3113   005264  075474                             EMT351
3114   005266  075526                             EHT1
3115   005270  075652                             EDT1
3116   005272  075742                             EFT1
3117
3118
3119                                    ;ERROR  352     PROGRAM TIMEOUT WHILE TESTING RMLA
3120   005274  075504                             EMT352
3121   005276  000000                             0
3122   005300  000000                             0
3123   005302  000000                             0
3124
3125
3126                                    ;ERROR  353     LOOK AHEAD TEST FAILS
3127   005304  075510                             EMT353
3128   005306  075646                             EHT353
3129   005310  075740                             EDT353
3130   005312  076026                             EFT353
3131
3132
3133                                    ;ERROR  354     BSE SHOULD NOT BE SET
3134   005314  075520                             EMT354
3135   005316  075526                             EHT1
3136   005320  075652                             EDT1
3137   005322  075742                             EFT1
3138
3139
3140                                    ;PUT ERROR TABLE HERE
3141                                    .SBTTL  ERROR TABLE USAGE
3142
3143                                    ;THE ERROR TABLE ABOVE CONSISTS OF FOUR WORD ENTRIES FOR EACH ERROR
3144                                    ;NUMBER, I.E.,
3145                                    ;
3146                                    ;       EMT - ERROR MESSAGE TABLE ADDRESS
3147                                    ;       EHT - ERROR HEADER TABLE ADDRESS
3148                                    ;       EDT - ERROR DATA TABLE ADDRESS
3149                                    ;       EFT - ERROR FORMAT TABLE ADDRESS
3150                                    ;
3151                                    ;THE EMT ENTRY IS THE ADDRESS OF THE TABLE OF ERROR MESSAGE STRINGS
3152                                    ;FOR THE PARTICULAR ERROR.  EACH ERROR MESSAGE TABLE LISTS THE ADDRESS
```

```
3153                    ;OF ONE OR MORE ERROR MESSAGE STRINGS WHICH ARE TO BE FORMATTED AND
3154                    ;TYPED BY THE ERROR TYPE SUBROUTINE.  IF THE EMT ENTRY IS ZERO, THERE IS
3155                    ;NO MESSAGE TO BE TYPED FOR THE ERROR.
3156                    ;
3157                    ;SIMILARLY, THE EHT, EDT, AND EFT ENTRIES ARE ADDRESSES OF TABLES
3158                    ;OF HEADER, DATA AND FORMAT INFORMATION FOR A GIVEN ERROR.  EACH ENTRY
3159                    ;IN THE ERROR HEADER TABLE MAY OR MAY NOT HAVE AN ASSOCIATED LINE OF
3160                    ;DATA, HOWEVER, EACH DATA LINE MUST HAVE AN ASSOCIATED FORMAT AND
3161                    ;HEADER. THAT IS, A HEADER LINE MAY BE PRINTED WITHOUT ANY DATA,
3162                    ;BUT A DATA LINE IS NOT PRINTED WITHOUT A HEADER, AND EACH DATA LINE
3163                    ;MUST ALSO HAVE A FORMAT.
3164                    ;
3165                    ;IN SUMMARY,
3166                    ;
3167                    ;       EACH NONZERO ENTRY IS THE ADDRESS OF A TABLE,
3168                    ;       EACH TABLE IS A LIST OF ADDRESSES WHICH DEFINES THE LOCATIONS
3169                    ;OF MESSAGE STRINGS, HEADERS, DATA OR FORMAT.
3170                    ;
3171
```

```
                                        .SBTTL  START OF PROGRAM

                                        START:

                                        ;CLEAR AND SETUP FOR TEST EXECUTION
005324  000240                                NOP
005326  000005                                RESET                   ;INITIALIZE THE SYSTEM
005330  013746  000300                        MOV     PR6,-(SP)       ;;PUT NEW PS ON STACK
005334  012746  005342                        MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
005340  000002                                RTI                     ;;POP NEW PC AND PS
005342                                  64$:

                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
005342  012706  001114                        MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
005346  005026                                CLR     (R6)+           ;;CLEAR MEMORY LOCATION
005350  022706  001154                        CMP     #$WR,R6 ;;DONE?
005354  001374                                BNE     .-6             ;;LOOP BACK IF NO
005356  012706  001100                        MOV     #STACK,SP       ;;SETUP THE STACK POINTER

                                        ;;INITIALIZE A FEW VECTORS
005362  012737  064600  000020                MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
005370  012737  000340  000022                MOV     #340,@#IOTVEC+2 ;;LEVEL 7
005376  012737  065174  000030                MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
005404  012737  000340  000032                MOV     #340,@#EMTVEC+2 ;;LEVEL 7
005412  012737  066734  000034                MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
005420  012737  000340  000036                MOV     #340,@#TRAPVEC+2;LEVEL 7
005426  012737  067042  000024                MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
005434  012737  000340  000026                MOV     #340,@#PWRVEC+2 ;;LEVEL 7
005442  013737  037042  037034                MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
005450  005037  001206                        CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
005454  005037  001210                        CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
005460  112737  000001  001131                MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
005466  012737  005466  001122                MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
005474  012737  005474  001124                MOV     #.,$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.
005502  013746  000004                        MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
005506  012737  005542  000004                MOV     #65$,@#ERRVEC   ;;SET UP ERROR VECTOR
005514  012737  177570  001154                MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
005522  012737  177570  001156                MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
005530  022777  177777  173416                CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
005536  001012                                BNE     67$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
005540  000403                                BR      66$             ;;BRANCH IF NO TIMEOUT
005542  012716  005550                  65$:  MOV     #66$,(SP)       ;;SET UP FOR TRAP RETURN
005546  000002                                RTI
005550  012737  000176  001154        66$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
005556  012737  000174  001156                MOV     #DISPREG,DISPLAY
005564  012637  000004                  67$:  MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

005570  005037  001230                        CLR     $PASS           ;;CLEAR PASS COUNT
005574  132737  000200  001243                BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
005602  001403                                BEQ     68$             ;;YES,USE NON-APT SWITCH
005604  012737  001244  001154                MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
005612                                  68$:
```

```
                                        .SBTTL  TYPE PROGRAM NAME
                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
005612  005227  177777                      INC     #-1             ;;FIRST TIME?
005616  001052                            BNE     69$             ;;BRANCH IF NO
005620  022737  037176  000042            CMP     #$ENDAD,@#42    ;;ACT-11?
005626  001446                            BEQ     69$             ;;BRANCH IF YES
005630  104401  005676                    TYPE    ,70$            ;;TYPE ASCIZ STRING
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
005634  005737  000042                    TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
005640  001012                            BNE     71$             ;;BRANCH IF YES
005642  123727  001242  000001            CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
005650  001406                            BEQ     71$             ;;BRANCH IF YES
005652  023727  001154  000176            CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
005660  001005                            BNE     72$             ;;BRANCH IF NO
005662  104407                            GTSWR                   ;;GET SOFT-SWR SETTINGS
005664  000403                            BR      72$
005666  112737  000001  001150  71$:      MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
005674                                  72$:
005674  000423                            BR      69$             ;;GET OVER THE ASCIZ
                                        ;;70$:  .ASCIZ  <CRLF>@RM03/RM02 FUNCTIONAL TEST, PART 3 @<CRLF>
005744                                  69$:


                                        ;FIND OUT IF PROGRAM IS RUNNING IN STANDALONE MODE
005744  122737  000077  000041            CMPB    #77,@#41        ;LOAD FROM THE RM03 FOR XXDP
005752  001003                            BNE     5$              ;BRANCH IF NOT
005754  104401  070565                    TYPE    ,XDPMG          ;THPE CHANGE PACK MESSAGE
005760  000000                            HALT                    ;THEN HALT
005762                                  5$:
005762  005737  000042                    TST     42              ;IS LOC 42 ZERO ??
005766  001003                            BNE     10$             ;NO - NOT IN STANDALONE
005770  105737  001242                    TSTB    $ENV            ;IS APT ENVIRONMENT ZERO ??
005774  001451                            BEQ     STANDALONE      ;YES - PROGRAM IN STANDALONE
005776                                  10$:


                                        ;PROGRAM NOT RUNNING IN STANDALONE - SEE IF SIZING IS ALLOWED
005776  132737  000200  001243  XSIZ:     BITB    #BIT7,$ENVM     ;SIZING ALLOWED ??
006004  001043                            BNE     20$             ;NO
                                        ;       MOV     #377,$DEVM      ;YES - SET DEVICE MAP FOR ALL DEVICES
006006  005037  001300                    CLR     $DEVM           ;CLEAR THE BIT MAP
006012  005001                            CLR     R1              ;START FROM THE DRIVE 0
006014  012704  000001                    MOV     #BIT0,R4        ;BIT MAP FOR DRIVE 0
006020  013700  001276                    MOV     $BASE,R0        ;LOAD THE BASE ADDRESS
006024  012760  000040  000010  15$:      MOV     #CLR,RMCS2(R0)  ;MASS BUS CLEAR
006032  010160  000010                    MOV     R1,RMCS2(R0)    ;LOAD THE BDRIVE ADDRESS
006036  016003  000010                    MOV     RMCS2(R0),R3    ;READ STATUS REG TO SEE NED BIT
006042  032703  010000                    BIT     #NED,R3         ;NED BIT SET ?
006046  001010                            BNE     16$             ;BRANCH IF SO
006050  016003  000000                    MOV     RMCS1(R0),R3    ;READ DVA BIT
006054  032703  004000                    BIT     #DVA,R3         ;DVA SET
006060  001403                            BEQ     16$             ;BRANCH IF NOT
006062  050437  001300                    BIS     R4,$DEVM        ;SET THE DEVICE MAP
006066  000405                            BR      17$             ;NOT TYPE THE NON-EXIST MESG
006070                                  16$:
006070  104401  070652                    TYPE    ,NOTEX          ;TYPE NOT EXIST DRIVE
006074  010146                            MOV     R1,-(SP)        ;DRIVE NUMBER
```

```
3284  006076  104403                        TYPOS
3285  006100     006                        .BYTE   6
3286  006101     000                        .BYTE   0
3287  006102  005201                17$:    INC     R1              ;INCREMENT THE DRIVE ADDRESS
3288  006104  006304                        ASL     R4              ;SET UP BIT MAP FOR THE NEXT DRIVE
3289  006106  022701  000007                CMP     #7,R1           ;ALL DRIVES ARE CHECKED /
3290  006112  103344                        BHIS    15$             ;BRANCH IF NOT
3291  006114                        20$:
3292
3293                                ;GO TO COMMON START CODE
3294  006114  000137  006746                JMP     CMNSTART
```

```
3295  006120                          STANDALONE:
3296  006120  004737  065404                JSR     PC,$TKINT       ;INITIALIZE CONSOLE
3297
3298                                  ;SEE IF THIS IS THE FIRST START AFTER PROGRAM WAS LOADED
3299  006124  005327  000001                DEC     #1              ;FIRST START ??
3300  006130  100024                        BPL     10$             ;YES !!
3301
3302
3303                                  ;SEE IF THE USER WANTS TO KEEP SAME DEVICES FOR TESTING
3304  006132  104401  070074          5$:   TYPE    ,CNSL00         ;MAINTAIN PREVIOUS PARAMETERS??
3305  006136  104411                        RDCHR                   ;GET RESPONSE
3306  006140  012637  001176                MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3307  006144  104401  001176                TYPE    ,$TMP1
3308  006150  123727  001176  000131        CMPB    $TMP1,#'Y       ;YES RESPONSE??
3309  006156  001002                        BNE     6$              ;NO!!
3310  006160  000137  007100                JMP     READY           ;KEEP PREVIOUS PARAMETERS
3311  006164  123727  001176  000116  6$:   CMPB    $TMP1,#'N       ;NO RESPONSE??
3312  006172  001420                        BEQ     20$             ;GET NEW PARAMETERS
3313  006174  104401  067605                TYPE    ,QSTMRK         ;NOT YES OR NO, TYPE "?"
3314  006200  000754                        BR      5$              ;RETRY
3315  006202                          10$:
3316
3317                                  ;SEE IF OPERATOR WANTS HELP FILE
3318  006202  104401  067607                TYPE    ,HELPQST        ;WANT HELP ??
3319  006206  104411                        RDCHR                   ;GET RESPONSE
3320  006210  012637  001176                MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
3321  006214  104401  001176                TYPE    ,$TMP1
3322  006220  123727  001176  000131        CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
3323  006226  001002                        BNE     20$             ;NO - DONT TYPE HELP
3324  006230  104401  107550                TYPE    ,HELP           ;YES - TYPE HELP TEXT
3325  006234                          20$:
3326
3327                                  ;SEE IF USER WANTS TO CHANGE RM03 UNIBUS ADDRESS
3328  006234  104401  067775                TYPE    ,UBUSQST        ;WANT TO CHANGE ADDRESS ??
3329  006240  104411                        RDCHR                   ;GET RESPONSE
3330  006242  012637  001176                MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
3331  006246  104401  001176                TYPE    ,$TMP1
3332  006252  104411                        RDCHR                   ;READ ONE MORE CHARACTER
3333  006254  005726                        TST     (SP)+           ;CLEAR THE STACK POINT
3334  006256  123727  001176  000131        CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
3335  006264  001137                        BNE     90$             ;NO !!
3336  006266                          30$:
3337
3338                                  ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, AND INTERRUPT VECTOR
3339  006266  104401  070133                TYPE    ,CNSL01         ;TYPE CURRENT BUS ADDRESS
3340  006272  013746  001276                MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
3341  006276  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3342  006300  104401  067576                TYPE    ,CLSPRN
3343  006304  104413                        RDOCT                   ;GET NEW BUS ADDRESS
3344  006306  012637  001176                MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
3345  006312  001412                        BEQ     50$             ;YES-SKIP TO NEXT ENTRY
3346  006314  022737  160000  001176        CMP     #160000,$TMP1   ;BASE ADDRESS IN I/O PAGE??
3347  006322  101403                        BLOS    40$             ;YES
3348  006324  104401  070160                TYPE    ,CNSL02         ;TYPE WARNING MESSAGE
3349  006330  000756                        BR      30$             ;RETRY
3350  006332  013737  001176  001276  40$:   MOV     $TMP1,$BASE     ;STORE NEW BUS ADDRESS
```

```
3351  006340  113737  001272  001176  50$:    MOVB    $VECT1,$TMP1    ;TYPE CURRENT VECTOR ADDRESS
3352  006346  105037  001177                  CLRB    $TMP1+1
3353  006352  104401  070241                  TYPE    ,CNSL03
3354  006356  013746  001176                  MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
3355  006362  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3356  006364     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
3357  006365     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3358  006366  104401  067576                  TYPE    ,CLSPRN
3359  006372  104413                          RDOCT                   ;GET NEW VECTOR ADDRESS
3360  006374  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN?
3361  006400  001412                          BEQ     70$             ;YES-SKIP TO NEXT ENTRY
3362  006402  022737  001000  001176          CMP     #1000,$TMP1     ;VECTOR ADDRESS < 1000??
3363  006410  101003                          BHI     60$             ;YES!!
3364  006412  104401  070271                  TYPE    ,CNSL04         ;TYPE WARNING MESSAGE
3365  006416  000750                          BR      50$             ;RETRY
3366  006420  113737  001176  001272  60$:    MOVB    $TMP1,$VECT1    ;STORE NEW VECTOR ADDRESS
3367  006426  113737  001273  001176  70$:    MOVB    $VECT1+1,$TMP1  ;TYPE CURRENT PRIORITY
3368  006434  006237  001176                  ASR     $TMP1
3369  006440  006237  001176                  ASR     $TMP1
3370  006444  006237  001176                  ASR     $TMP1
3371  006450  006237  001176                  ASR     $TMP1
3372  006454  006237  001176                  ASR     $TMP1
3373  006460  105037  001177                  CLRB    $TMP1+1
3374  006464  104401  070345                  TYPE    ,CNSL05
3375  006470  013746  001176                  MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
3376  006474  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3377  006476     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
3378  006477     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3379  006500  104401  067576                  TYPE    ,CLSPRN
3380  006504  104413                          RDOCT                   ;GET NEW PRIORITY
3381  006506  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
3382  006512  001424                          BEQ     90$             ;YES-SKIP TO NEXT ENTRY
3383  006514  023727  001176  000007          CMP     $TMP1,#7        ;LEGAL PRIORITY??
3384  006522  002403                          BLT     80$             ;YES!!
3385  006524  104401  070401                  TYPE    ,CNSL06         ;TYPE WARNING MESSAGE
3386  006530  000736                          BR      70$             ;RETRY
3387  006532                          80$:                            ;STORE NEW PRIORITY
3388  006532  006337  001176                  ASL     $TMP1
3389  006536  006337  001176                  ASL     $TMP1
3390  006542  006337  001176                  ASL     $TMP1
3391  006546  006337  001176                  ASL     $TMP1
3392  006552  006337  001176                  ASL     $TMP1
3393  006556  113737  001176  001273          MOVB    $TMP1,$VECT1+1
3394  006564                          90$:
3395
3396                                  ;DIALOGUE TO INPUT DEVICE NUMBERS
3397  006564  005037  001300                  CLR     $DEVM           ;CLEAR DEVICE MAP
3398  006570  104401  070426                  TYPE    ,CNSL07         ;TYPE INPUT INSTRUCTIONS
3399  006574  104401  067601                  TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
3400  006600  104411                          RDCHR                   ;GET RESPONSE
3401  006602  012637  001176                  MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3402  006606  104401  001176                  TYPE    ,$TMP1
3403  006612  023727  001176  000101          CMP     $TMP1,#'A       ;TEST ALL DRIVES??
3404  006620  001021                          BNE     110$            ;NO
3405  006622  000137  005776                  JMP     XSIZ            ;AUTO SIZE
3406  006626  012737  000377  001300          MOV     #377,$DEVM      ;TEST ALL DEVICES
```

```
3407  006634  000444                    BR      140$            ;SKIP TO NEXT ENTRY
3408  006636  104401  067601    100$:   TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
3409  006642  104411                    RDCHR                   ;GET RESPONSE
3410  006644  012637  001176            MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3411  006650  104401  001176            TYPE    ,$TMP1
3412  006654  023727  001176  000015    CMP     $TMP1,#CR       ;CARRIAGE RETURN??
3413  006662  001431                    BEQ     140$
3414  006664  023727  001176  000060    CMP     $TMP1,#'0       ;NUMBER < 0??
3415  006672  002404                    BLT     120$            ;YES!!
3416  006674  023727  001176  000067    CMP     $TMP1,#'7       ;NUMBER > 7??
3417  006702  003403                    BLE     130$            ;NO!!
3418  006704  104401  067605    120$:   TYPE    ,QSTMRK         ;TYPE "?"
3419  006710  000752                    BR      100$            ;RETRY
3420  006712  013701  001176    130$:   MOV     $TMP1,R1        ;R1=DRIVE NUMBER
3421  006716  042701  177770            BIC     #^C7,R1
3422  006722  116102  070776            MOVB    ATNTBL(R1),R2           ;DECODE DEVICE NUMBER
3423  006726  042702  177400            BIC     #^C377,R2               ;CLEAR UNUSED BITS
3424  006732  050237  001300            BIS     R2,$DEVM                ;SET DEVICE # IN MAP
3425  006736  122737  000377  001300    CMPB    #377,$DEVM      ;DONE ??
3426  006744  101334                    BHI     100$            ;NO
3427  006746                    140$:
3428
```

```
 3429  006746                                CMNSTART:
 3430
 3431                                        ;ASSEMBLE TEST QUE FROM DEVICE MAP
 3432  006746  013700  001300                       MOV     $DEVM,R0          ;R0 = DEVICE MAP
 3433  006752  012701  001452                       MOV     #TSTQUE+2,R1      ;R1 = ADDRESS OF FIRST ENTRY IN QUE
 3434  006756  010137  001450                       MOV     R1,TSTQUE         ;INITIALIZE ENTRY POINTER
 3435  006762  012702  000001                       MOV     #1,R2             ;R2 = DEVICE POINTER
 3436  006766  005003                             CLR     R3                ;R3 = DEVICE NUMBER
 3437  006770  030200                       10$:  BIT     R2,R0             ;IS THIS DEVICE IN MAP ??
 3438  006772  001406                             BEQ     20$               ;NO !!
 3439  006774  010311                             MOV     R3,(R1)           ;YES - ENTER DEVICE NUMBER IN QUE
 3440  006776  116361  070776  000001             MOVB    ATNTBL(R3),1(R1)  ;ENTER ATTENTION BIT IN QUE
 3441  007004  062701  000002                       ADD     #2,R1             ;ADVANCE ENTRY POINTER
 3442  007010  006302                       20$:  ASL     R2                ;ADVANCE DEVICE POINTER
 3443  007012  105702                             TSTB    R2                ;DONE ALL DEVICES ??
 3444  007014  001402                             BEQ     25$               ;YES
 3445  007016  005203                             INC     R3                ;ADVANCE DEVICE NUMBER
 3446  007020  000763                             BR      10$               ;ENTER NEXT DEVICE
 3447  007022  005011                       25$:  CLR     (R1)              ;TERMINATE TEST QUE
 3448
 3449                                        ;SIZE FOR CLOCK
 3450  007024  004737  044224                       JSR     PC,SIZCLK         ;SEE IF CLOCK PRESENT
 3451  007030  000403                             BR      40$               ;YES - CLOCK IS PRESENT
 3452  007032  104000                       30$:  ERROR                     ;NO CLOCK
 3453  007034  000000                             HALT
 3454  007036  000775                             BR      30$
 3455  007040                                40$:
 3456                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
 3457  007040  005737  000042                       TST     @#42              ;;ARE WE RUNNING UNDER XXDP/ACT?
 3458  007044  001012                             BNE     64$               ;;BRANCH IF YES
 3459  007046  123727  001242  000001             CMPB    $ENV,#1           ;;ARE WE RUNNING UNDER APT?
 3460  007054  001406                             BEQ     64$               ;;BRANCH IF YES
 3461  007056  023727  001154  000176             CMP     SWR,#SWREG        ;;SOFTWARE SWITCH REG SELECTED?
 3462  007064  001005                             BNE     65$               ;;BRANCH IF NO
 3463  007066  104407                             GTSWR                     ;;GET SOFT-SWR SETTINGS
 3464  007070  000403                             BR      65$
 3465  007072  112737  000001  001150  64$:       MOVB    #1,$AUTOB         ;;SET AUTO-MODE INDICATOR
 3466  007100                                65$:
 3467  007100  000240                       READY:  NOP                       ;READY TO START TEST
 3468  007102  005037  001326                       CLR     CTLFG             ;CLEAR THE CONTROL-C FLAG
 3469  007106  004737  065404                       JSR     PC,$TKINT         ;INITIALIZE TTY
 3470  007112  013746  000300                       MOV     PR6,-(SP)         ;;PUT NEW PS ON STACK
 3471  007116  012746  007124                       MOV     #64$,-(SP)        ;;PUT NEW PC ON  STACK
 3472  007122  000002                             RTI                       ;;POP NEW PC AND PS
 3473  007124                                64$:
 3474  007124  117737  172320  001234             MOVB    @TSTQUE,$UNIT     ;LOAD UNIT NUMBER
 3475  007132  005037  001474                       CLR     MEDENB            ;CLEAR MEDIA ENABLE
```

```
                                ;;*****************************************************************
                                ;*TEST 1         CONTROLLER ACCESS TEST
                                ;;*****************************************************************
007136                          TST1:
007136  000240                          NOP                             ;START OF TEST
007140  012737  007154  001122          MOV     #1$,$LPADR
007146  012737  007154  001124          MOV     #1$,$LPERR
007154                          1$:
007154  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
007160  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
007164  013701  001450                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
007170  012737  000001  001226          MOV     #1,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
007176  005001                          CLR     R1
007200  013746  000004                  MOV     ERRVEC,-(SP)            ;;PUSH ERRVEC ON STACK
007204  013746  000006                  MOV     ERRVEC+2,-(SP)                  ;;PUSH ERRVEC+2 ON STACK
007210  012737  007302  000004          MOV     #3$,ERRVEC
007216  012737  000300  000006          MOV     #PR6,ERRVEC+2
                                
007224  110160  000001                  MOVB    R1,RMCS1+1(R0)          ;MOVE HI BYTE TO RMCS1
007230  010160  000002                  MOV     R1,RMWC(R0)             ;MOVE WORD COUNT REGISTER
007234  016002  000002                  MOV     RMWC(R0),R2
007240  010160  000004                  MOV     R1,RMBA(R0)             ;MOVE BUS ADDRESS REGISTER
007244  016002  000004                  MOV     RMBA(R0),R2
007250  010160  000010                  MOV     R1,RMCS2(R0)            ;MOVE CONTROL STATUS REGISTER
007254  016002  000010                  MOV     RMCS2(R0),R2
007260  010160  000022                  MOV     R1,RMDB(R0)             ;MOVE DATA BUFFER
007264  016002  000022                  MOV     RMDB(R0),R2
007270  012637  000006                  MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
007274  012637  000004                  MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
007300  000415                          BR      7$                      ;NO BUS TIMEOUT OCCURRED
                                
007302  022626                  3$:     CMP     (SP)+,(SP)+             ;ADJUST STACK
007304  012637  000006                  MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
007310  012637  000004                  MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
007314  104110                          ERROR   110                     ;CANNOT ACCESS MASSBUS CONTROLLER
007316  005737  000042                  TST     42                      ;STAND ALONE MODE??
007322  001002                          BNE     5$                      ;NO!!
007324  000137  005324                  JMP     START                   ;YES-GO GET $BASE
007330  000137  037006          5$:     JMP     $EOP                    ;GO TO END OF PASS HANDLER
007334                          7$:
                                
                                ;;*****************************************************************
                                ;*TEST 2         DEVICE AVAILABLE TEST
                                
                                ;;*****************************************************************
007334                          TST2:
007334  000004                          SCOPE                           ;SCOPE CALL
007336  000240                          NOP                             ;START OF TEST
007340  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
007344  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
007350  013701  001450                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
007354  012737  000002  001226          MOV     #2,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX

007362  004737  052700                  JSR     PC,CNTCLR
007366  000404                          BR      2$      ;GO TO 2$ IF NO ERROR
007370  000240                          NOP                             ;RETURN HERE IF ERROR
```

```
3532  007372  104000                        ERROR                   ;ERROR NUMBER DEFINED BY SUB
3533  007374  000137  007514                JMP     7$      ;GO TO 7$ IF ERROR WAS FOUND
3534  007400                        2$:
3535  007400  013746  000004                MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
3536  007404  013746  000006                MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
3537  007410  012737  007500  000004        MOV     #5$,ERRVEC
3538  007416  013737  000300  000006        MOV     PR6,ERRVEC+2
3539
3540  007424  016037  000000  001176        MOV     RMCS1(R0),$TMP1 ;GET DVA STATUS
3541  007432  016037  000010  001174        MOV     RMCS2(R0),$TMP0 ;GET NED STATUS
3542  007440  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
3543  007444  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
3544  007450  032737  010000  001174        BIT     #NED,$TMP0      ;NONEXISTENT DEVICE??
3545  007456  001402                        BEQ     3$              ;NO!!
3546  007460  104111                        ERROR   111             ;NONEXISTENT DEVICE
3547  007462  000414                        BR      7$
3548  007464  032737  004000  001176  3$:   BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
3549  007472  001012                        BNE     9$              ;YES!!
3550  007474  104112                        ERROR   112             ;DEVICE NOT AVAILABLE
3551  007476  000406                        BR      7$
3552
3553  007500  022626                5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
3554  007502  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
3555  007506  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
3556  007512  104113                        ERROR   113             ;BUS TIMEOUT (04 TRAP)
3557  007514  000137  036752        7$:     JMP     $EOSP
3558
3559  007520                        9$:
3560
3561                                ;;*****************************************************************
3562                                ;*TEST 3        DRIVE TYPE TEST
3563
3564                                ;;*****************************************************************
3565  007520                        TST3:
3566  007520  000004                        SCOPE                   ;SCOPE CALL
3567  007522  000240                        NOP                     ;START OF TEST
3568  007524  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3569  007530  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3570  007534  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3571  007540  012737  000003  001226        MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3572
3573  007546  004737  052700                JSR     PC,CNTCLR
3574  007552  000404                        BR      2$      ;GO TO 2$ IF NO ERROR
3575  007554  000240                        NOP                     ;RETURN HERE IF ERROR
3576  007556  104000                        ERROR                   ;ERROR NUMBER DEFINED BY SUB
3577  007560  000137  007676                JMP     4$      ;GO TO 4$ IF ERROR WAS FOUND
3578  007564                        2$:
3579  007564  112737  000026  001506        MOVB    #RMDT,GETINX    ;SETUP GET INDEX TABLE
3580  007572  112737  000200  001507        MOVB    #200,GETINX+1
3581  007600  012737  007702  001356        MOV     #5$,RMDTI       ;RMDT INPUT BUFFER = 5$
3582  007606  004737  043536                JSR     PC,GET          ;GO GET DRIVE TYPE
3583  007612  000402                        BR      3$      ;GO TO 3$ IF NO ERROR
3584  007614  000240                        NOP                     ;RETURN HERE IF ERROR
3585  007616  104000                        ERROR                   ;ERROR NUMBER SPECIFIED BY GET
3586  007620  022737  020024  001356  3$:   CMP     #SNGPRT,RMDTI   ;SINGLE PORT RM03??
3587  007626  001425                        BEQ     5$              ;YES!!
```

```
3588  007630  022737  024024  001356          CMP     #DULPRT,RMDTI   ;DUAL PORT RM03??
3589  007636  001421                          BEQ     5$              ;YES!!
3590  007640  022737  020025  001356          CMP     #SNGPRT!BIT0,RMDTI      ;SINGLE PROT RM02 ?
3591  007646  001415                          BEQ     5$              ;
3592  007650  022737  024025  001356          CMP     #DULPRT!BIT0,RMDTI      ;DUAL PORT RM02 ?
3593  007656  001411                          BEQ     5$              ;
3594  007660  012737  020024  001176          MOV     #SNGPRT,$TMP1
3595  007666  012737  024024  001200          MOV     #DULPRT,$TMP2
3596  007674  104114                          ERROR   114             ;NOT AN RM03
3597  007676  000137  036752          4$:     JMP     $EOSP           ;GO TO SUBPASS HANDLER.
3598
3599  007702                          5$:
3600                                  ;;*****************************************************************
3601                                  ;*TEST 4         WRITE, READ ZEROS
3602                                  ;;*****************************************************************
3603  007702                          TST4:
3604  007702  000004                          SCOPE                   ;SCOPE CALL
3605  007704  000240                          NOP                     ;START OF TEST
3606  007706  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3607  007712  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3608  007716  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3609  007722  012737  000004  001226          MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3610
3611                                  ;;*****************************************************************
3612                                  ;LOOP #1        FORMAT,WRITE,READ
3613
3614  007730                          10$:
3615
3616                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
3617  007730  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
3618  007734  054130                          .WORD   054130  ;TASK DESCRIPTOR
3619  007736  000404                          BR      20$             ;GO TO 20$ IF NO ERROR
3620  007740  000240                          NOP                     ;RETURN HERE IF ERROR
3621  007742  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3622  007744  000137  010724                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
3623  007750                          20$:
3624
3625                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
3626  007750  012737  000000  001434          MOV     #0,RMDC0        ;CYLINDER = 0
3627  007756  012737  000000  001406          MOV     #0,RMDA0        ;TRACK = 0, SECTOR = 0
3628  007764  012737  107550  001404          MOV     #BUFONE,RMBA0   ;BUS ADDRESS
3629  007772  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT = 1 SECTOR
3630  010000  012737  010000  001432          MOV     #FMT16,RMOF0    ;16 BIT FORMAT
3631  010006  012737  000063  001400          MOV     #WH!GO,RMCS10   ;WRITE HEADER AND DATA COMMAND
3632
3633                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
3634
3635  010014  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
3636  010020  000405                          BR      25$             ;GO TO 25$ IF NO ERROR
3637  010022  104401  067466                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
3638  010026  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
3639  010030  000137  010724                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
3640  010034                          25$:
3641  010034  012737  071110  001174          MOV     #ZEROS,$TMP0    ;STARTING ADDRESS OF PATTERN
3642  010042  012737  000001  001176          MOV     #1,$TMP1        ;RANGE OF PATTERN
3643  010050  004737  042640                  JSR     PC,GENBUF       ;GO GENERATE BUFFER FOR FORMAT
```

```
                                ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
010054  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
010060  112722  000034                  MOVB    #RMDC,(R2)+
010064  112722  000006                  MOVB    #RMDA,(R2)+
010070  112722  000004                  MOVB    #RMBA,(R2)+
010074  112722  000002                  MOVB    #RMWC,(R2)+
010100  112722  000032                  MOVB    #RMOF,(R2)+
010104  112722  000000                  MOVB    #RMCS1,(R2)+
010110  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
010114                          30$:

                                ;FORMAT THE DRIVE
010114  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
010120  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
010122  000240                          NOP                     ;RETURN HERE IF ERROR
010124  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
010126  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
010132                          40$:

                                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
010132  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR THE FORMAT TO COMPLETE
010136  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR FORMAT OPERATION
010142  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
010146  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
010150  000240                          NOP                     ;RETURN HERE IF ERROR
010152  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
010154  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
010160                          50$:

                                ;VERIFY NO ERRORS DURING FORMAT
010160  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
010164  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
010166  000240                          NOP                     ;RETURN HERE IF ERROR
010170  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
010172  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
010174  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
010200                          60$:

                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
010200  012737  010216  001122          MOV     #70$,$LPADR
010206  012737  010216  001124          MOV     #70$,$LPERR
010214  000410                          BR      80$                     ;SKIP TO WRITE OPERATION

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

010216                          70$:


                                ;PREPARE DEVICE FOR WRITE OPERATION
010216  004737  040020                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
```

```
3700  010222  054130                        .WORD   054130  ;TASK DESCRIPTOR
3701  010224  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
3702  010226  000240                        NOP                     ;RETURN HERE IF ERROR
3703  010230  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3704  010232  000137  010724                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3705  010236                        80$:
3706
3707
3708  010236                        120$:
3709
3710                                ;WRITE DATA TO THE DRIVE
3711  010236  012737  107554  001404        MOV     #BUFONE+4,RMBAO         ;MOVE MEMORY ADDRESS
3712  010244  012737  177400  001402        MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
3713  010252  012737  000061  001400        MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
3714  010260  012702  001535                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
3715  010264  112722  000006                MOVB    #RMDA,(R2)+
3716  010270  112722  000034                MOVB    #RMDC,(R2)+
3717  010274  112722  000032                MOVB    #RMOF,(R2)+
3718  010300  112722  000004                MOVB    #RMBA,(R2)+
3719  010304  112722  000002                MOVB    #RMWC,(R2)+
3720  010310  112722  000000                MOVB    #RMCS1,(R2)+
3721  010314  112722  000200                MOVB    #200,(R2)+              ;TERMINATE TABLE
3722
3723  010320  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
3724  010324  000404                        BR      130$                    ;GO TO 130$ IF NO ERROR
3725  010326  000240                        NOP                     ;RETURN HERE IF ERROR
3726  010330  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
3727  010332  000137  010724                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3728  010336                        130$:
3729
3730                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
3731  010336  004737  043452                JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
3732
3733                                ;WAIT FOR WRITE COMMAND TO COMPLETE
3734  010342  004737  044346                JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
3735
3736                                ;GO READ STATUS FOR WRITE COMMAND
3737  010346  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
3738  010352  000404                        BR      140$                    ;GO TO 140$ IF NO ERROR
3739  010354  000240                        NOP                     ;RETURN HERE IF ERROR
3740  010356  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
3741  010360  000137  010724                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3742  010364                        140$:
3743
3744                                ;CHECK FOR ERRORS DURING WRITE OPERATION
3745  010364  004737  044532                JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
3746  010370  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
3747  010372  000240                        NOP                     ;RETURN HERE IF ERROR
3748  010374  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
3749  010376  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
3750  010400  000137  010724                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3751  010404                        150$:
3752  010404  004737  057036                JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
3753  010410  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
3754  010412  000240                        NOP                     ;RETURN HERE IF ERROR
3755  010414  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
```

```
3756  010416  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3757  010420  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3758  010424                          160$:
3759
3760  010424                          170$:
3761  010424  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
3762  010430  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
3763  010432  000240                          NOP                     ;RETURN HERE IF ERROR
3764  010434  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
3765  010436  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3766  010440  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3767  010444                          180$:
3768
3769                                  ;CHANGE LOOP ADDRESSES
3770  010444  012737  010462  001122          MOV     #190$,$LPADR
3771  010452  012737  010462  001124          MOV     #190$,$LPERR
3772  010460  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
3773
3774                                  ;;*****************************************************************
3775                                  ;LOOP #3        READ
3776
3777  010462                          190$:
3778
3779                                  ;PREPARE DEVICE FOR READ OPERATION
3780  010462  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
3781  010466  054130                          .WORD   054130  ;TASK DESCRIPTOR
3782  010470  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
3783  010472  000240                          NOP                     ;RETURN HERE IF ERROR
3784  010474  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3785  010476  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3786  010502                          200$:
3787
3788  010502                          240$:
3789
3790                                  ;READ DATA FROM DEVICE
3791  010502  012737  110560  001404          MOV     #BUFTWO+4,RMBA0         ;CHANGE MEMORY ADDRESS
3792  010510  012737  000071  001400          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
3793  010516  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
3794  010522  112722  000006                  MOVB    #RMDA,(R2)+
3795  010526  112722  000032                  MOVB    #RMOF,(R2)+
3796  010532  112722  000034                  MOVB    #RMDC,(R2)+
3797  010536  112722  000004                  MOVB    #RMBA,(R2)+
3798  010542  112722  000002                  MOVB    #RMWC,(R2)+
3799  010546  112722  000000                  MOVB    #RMCS1,(R2)+
3800  010552  112712  000200                  MOVB    #200,(R2)
3801
3802  010556  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
3803  010562  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
3804  010564  000240                          NOP                     ;RETURN HERE IF ERROR
3805  010566  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
3806  010570  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3807  010574                          250$:
3808
3809                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
3810  010574  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
3811
```

```
3812                                    ;WAIT FOR READ OPERATION TO COMPLETE
3813  010600  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
3814
3815                                    ;GO READ STATUS FOR READ OPERATION
3816  010604  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
3817  010610  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
3818  010612  000240                          NOP                     ;RETURN HERE IF ERROR
3819  010614  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
3820  010616  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3821  010622                          260$:
3822
3823                                    ;CHECK FOR ERRORS DURING READ OPERATION
3824  010622  004737  044532                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
3825  010626  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
3826  010630  000240                          NOP                     ;RETURN HERE IF ERROR
3827  010632  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
3828  010634  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3829  010636  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3830  010642                          270$:
3831  010642  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
3832  010646  000405                          BR      280$                    ;GO TO 280$ IF NO ERROR
3833  010650  000240                          NOP                     ;RETURN HERE IF ERROR
3834  010652  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3835  010654  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3836  010656  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3837  010662                          280$:
3838
3839  010662                          290$:
3840  010662  004737  045364                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
3841  010666  000405                          BR      300$                    ;GO TO 300$ IF NO ERROR
3842  010670  000240                          NOP                     ;RETURN HERE IF ERROR
3843  010672  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
3844  010674  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3845  010676  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3846  010702                          300$:
3847  010702  004737  043104                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
3848  010706  107554                          .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
3849  010710  110560                          .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
3850  010712  000404                          BR      310$                    ;GO TO 310$ IF NO ERROR
3851  010714  000240                          NOP                     ;RETURN HERE IF ERROR
3852  010716  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
3853  010720  000137  010724                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
3854  010724                          310$:
3855
3856  010724                          350$:
3857                                    ;;*****************************************************************
3858                                    ;*TEST 5         WRITE, WRITE CHECK ZEROS
3859                                    ;;*****************************************************************
3860  010724                          TST5:
3861  010724  000004                          SCOPE                   ;SCOPE CALL
3862  010726  000240                          NOP                     ;START OF TEST
3863  010730  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3864  010734  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3865  010740  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3866  010744  012737  000005  001226          MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3867
```

```
                                ;;*****************************************************************
                                ;LOOP #1        FORMAT,WRITE,WRITE CHECK DATA

010752                          10$:

                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
010752  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
010756  054130                  .WORD   054130  ;TASK DESCRIPTOR
010760  000404                  BR      20$                     ;GO TO 20$ IF NO ERROR
010762  000240                  NOP                     ;RETURN HERE IF ERROR
010764  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
010766  000137  011716          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
010772                          20$:

                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
010772  012737  000000  001434        MOV     #0,RMDC0                ;CYLINDER = 0
011000  012737  000000  001406        MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
011006  012737  107550  001404        MOV     #BUFONE,RMBA0           ;BUS ADDRESS
011014  012737  177376  001402        MOV     #<^C<2+256.>+1>,RMWC0           ;WORD COUNT = 1 SECTOR
011022  012737  010000  001432        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
011030  012737  000063  001400        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND

                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE

011036  004737  040734                JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
011042  000405                  BR      25$                     ;GO TO 25$ IF NO ERROR
011044  104401  067466          TYPE    ,SCTMSG                 ;TYPE BAD SECTOR MESSAGE
011050  104000                  ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
011052  000137  011716          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
011056                          25$:
011056  012737  071110  001174        MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
011064  012737  000001  001176        MOV     #1,$TMP1                ;RANGE OF PATTERN
011072  004737  042640                JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT

                                ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
011076  012702  001535        MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
011102  112722  000034        MOVB    #RMDC,(R2)+
011106  112722  000006        MOVB    #RMDA,(R2)+
011112  112722  000004        MOVB    #RMBA,(R2)+
011116  112722  000002        MOVB    #RMWC,(R2)+
011122  112722  000032        MOVB    #RMOF,(R2)+
011126  112722  000000        MOVB    #RMCS1,(R2)+
011132  112712  000200        MOVB    #200,(R2)               ;TERMINATE TABLE
011136                          30$:

                                ;FORMAT THE DRIVE
011136  004737  044006        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
011142  000404                  BR      40$                     ;GO TO 40$ IF NO ERROR
011144  000240                  NOP                     ;RETURN HERE IF ERROR
011146  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
011150  000137  011716          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
011154                          40$:

                                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
011154  004737  043452        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
```

```
                                        ;WAIT FOR THE FORMAT TO COMPLETE
011160  004737  044346                          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                        ;GO READ STATUS FOR FORMAT OPERATION
011164  004737  043536                          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
011170  000404                                  BR      50$                     ;GO TO 50$ IF NO ERROR
011172  000240                                  NOP                     ;RETURN HERE IF ERROR
011174  104000                                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
011176  000137  011716                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
011202                                  50$:

                                        ;VERIFY NO ERRORS DURING FORMAT
011202  004737  057036                          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
011206  000405                                  BR      60$                     ;GO TO 60$ IF NO ERROR
011210  000240                                  NOP                     ;RETURN HERE IF ERROR
011212  104000                                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
011214  004736                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
011216  000137  011716                          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
011222                                  60$:

                                        ;MOVE LOOP ADDRESSES TO NEXT OPERATION
011222  012737  011240  001122                  MOV     #70$,$LPADR
011230  012737  011240  001124                  MOV     #70$,$LPERR
011236  000410                                  BR      80$                     ;SKIP TO WRITE OPERATION

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

011240                                  70$:

                                        ;PREPARE DEVICE FOR WRITE OPERATION
011240  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
011244  054130                          .WORD   054130  ;TASK DESCRIPTOR
011246  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
011250  000240                          NOP                     ;RETURN HERE IF ERROR
011252  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
011254  000137  011716                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
011260                                  80$:


011260                                  120$:

                                        ;WRITE DATA TO THE DRIVE
011260  012737  107554  001404          MOV     #BUFONE+4,RMBA0         ;CHANGE MEMORY ADDRESS
011266  012737  177400  001402          MOV     #<^C256.+1>,RMWC0       ;CHANGE WORD COUNT
011274  012737  000061  001400          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
011302  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
011306  112722  000006                  MOVB    #RMDA,(R2)+
011312  112722  000034                  MOVB    #RMDC,(R2)+
011316  112722  000032                  MOVB    #RMOF,(R2)+
011322  112722  000004                  MOVB    #RMBA,(R2)+
011326  112722  000002                  MOVB    #RMWC,(R2)+
011332  112722  000000                  MOVB    #RMCS1,(R2)+
011336  112722  000200                  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        WRITE CHECK DATA

                                190$:
```

```
                                ;PREPARE DEVICE FOR WRITE CHECK DATA OPERATION
011504  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
011510  054130                          .WORD   054130  ;TASK DESCRIPTOR
011512  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
011514  000240                          NOP                     ;RETURN HERE IF ERROR
011516  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
011520  000137  011716                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
011524                          200$:

011524                          240$:

                                ;WRITE CHECK DATA DATA FROM DEVICE
011524  012737  000051  001400          MOV     #WCD!GO,RMCS10          ;WRITE CHECK DATA DATA COMMAND
011532  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
011536  112722  000006                  MOVB    #RMDA,(R2)+
011542  112722  000032                  MOVB    #RMOF,(R2)+
011546  112722  000034                  MOVB    #RMDC,(R2)+
011552  112722  000004                  MOVB    #RMBA,(R2)+
011556  112722  000002                  MOVB    #RMWC,(R2)+
011562  112722  000000                  MOVB    #RMCS1,(R2)+
011566  112712  000200                  MOVB    #200,(R2)

011572  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
011576  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
011600  000240                          NOP                     ;RETURN HERE IF ERROR
011602  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
011604  000137  011716                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
011610                          250$:

                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
011610  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR WRITE CHECK DATA OPERATION TO COMPLETE
011614  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
011620  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
011624  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
011626  000240                          NOP                     ;RETURN HERE IF ERROR
011630  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
011632  000137  011716                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
011636                          260$:

                                ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
011636  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
011642  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
011644  000240                          NOP                     ;RETURN HERE IF ERROR
011646  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
011650  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
011652  000137  011716                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
011656                          270$:
011656  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
011662  000405                          BR      280$                    ;GO TO 280$ IF NO ERROR
011664  000240                          NOP                     ;RETURN HERE IF ERROR
011666  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
011670  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
4092  011672  000137  011716                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4093  011676                            280$:
4094
4095  011676                            290$:
4096  011676  004737  045364                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4097  011702  000405                            BR      300$                    ;GO TO 300$ IF NO ERROR
4098  011704  000240                            NOP                     ;RETURN HERE IF ERROR
4099  011706  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4100  011710  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4101  011712  000137  011716                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4102  011716                            300$:
4103
4104  011716                            350$:
4105                                    ;;*****************************************************************
4106                                    ;*TEST 6        WRITE, WRITE CHECK ZEROS W/ WCE ERROR
4107                                    ;;*****************************************************************
4108  011716                            TST6:
4109  011716  000004                            SCOPE                   ;SCOPE CALL
4110  011720  000240                            NOP                     ;START OF TEST
4111  011722  012706  001100                    MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4112  011726  013700  001276                    MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4113  011732  013701  001450                    MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4114  011736  012737  000006  001226            MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4115
4116                                    ;;*****************************************************************
4117                                    ;LOOP #1        FORMAT,WRITE,WRITE CHECK DATA
4118
4119  011744                            10$:
4120
4121                                    ;PREPARE THE DEVICE FOR FORMAT OPERATION
4122  011744  004737  040020                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4123  011750  054130                            .WORD   054130  ;TASK DESCRIPTOR
4124  011752  000404                            BR      20$                     ;GO TO 20$ IF NO ERROR
4125  011754  000240                            NOP                     ;RETURN HERE IF ERROR
4126  011756  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4127  011760  000137  013134                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4128  011764                            20$:
4129
4130                                    ;LOAD PARAMETERS AND GENERATE DATA BUFFER
4131  011764  012737  000000  001434            MOV     #0,RMDC0                ;CYLINDER = 0
4132  011772  012737  000000  001406            MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
4133  012000  012737  107550  001404            MOV     #BUFONE,RMBA0           ;BUS ADDRESS
4134  012006  012737  177376  001402            MOV     #<^C<2+256.>+1>,RMWC0           ;WORD COUNT = 1 SECTOR
4135  012014  012737  010000  001432            MOV     #FMT16,RM0F0            ;16 BIT FORMAT
4136  012022  012737  000063  001400            MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
4137
4138                                    ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
4139
4140  012030  004737  040734                    JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
4141  012034  000405                            BR      25$                     ;GO TO 25$ IF NO ERROR
4142  012036  104401  067466                    TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
4143  012042  104000                            ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
4144  012044  000137  013134                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4145  012050                            25$:
4146  012050  012737  071110  001174            MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
4147  012056  012737  000001  001176            MOV     #1,$TMP1                ;RANGE OF PATTERN
```

```
4148  012064  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
4149
4150                                          ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
4151  012070  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
4152  012074  112722  000034                  MOVB    #RMDC,(R2)+
4153  012100  112722  000006                  MOVB    #RMDA,(R2)+
4154  012104  112722  000004                  MOVB    #RMBA,(R2)+
4155  012110  112722  000002                  MOVB    #RMWC,(R2)+
4156  012114  112722  000032                  MOVB    #RMOF,(R2)+
4157  012120  112722  000000                  MOVB    #RMCS1,(R2)+
4158  012124  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
4159  012130                                30$:
4160
4161                                          ;FORMAT THE DRIVE
4162  012130  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4163  012134  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
4164  012136  000240                          NOP                     ;RETURN HERE IF ERROR
4165  012140  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4166  012142  000137  013134                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4167  012146                                40$:
4168
4169                                          ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
4170  012146  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
4171
4172                                          ;WAIT FOR THE FORMAT TO COMPLETE.
4173  012152  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
4174
4175                                          ;GO READ STATUS FOR FORMAT OPERATION
4176  012156  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4177  012162  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
4178  012164  000240                          NOP                     ;RETURN HERE IF ERROR
4179  012166  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4180  012170  000137  013134                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4181  012174                                50$:
4182
4183                                          ;VERIFY NO ERRORS DURING FORMAT
4184  012174  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
4185  012200  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
4186  012202  000240                          NOP                     ;RETURN HERE IF ERROR
4187  012204  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4188  012206  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4189  012210  000137  013134                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4190  012214                                60$:
4191
4192                                          ;MOVE LOOP ADDRESSES TO NEXT OPERATION
4193  012214  012737  012232  001122          MOV     #70$,$LPADR
4194  012222  012737  012232  001124          MOV     #70$,$LPERR
4195  012230  000410                          BR      80$                     ;SKIP TO WRITE OPERATION
4196
4197                                          ;;*****************************************************************
4198                                          ;LOOP #2        WRITE,WRITE CHECK DATA
4199
4200  012232                                70$:
4201
4202
4203                                          ;PREPARE DEVICE FOR WRITE OPERATION
```

```
4204  012232  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4205  012236  054130                  .WORD   054130  ;TASK DESCRIPTOR
4206  012240  000404                  BR      80$                     ;GO TO 80$ IF NO ERROR
4207  012242  000240                  NOP                     ;RETURN HERE IF ERROR
4208  012244  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4209  012246  000137  013134          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4210  012252                  80$:
4211
4212
4213  012252                  120$:
4214
4215                          ;WRITE DATA TO THE DRIVE
4216  012252  012737  177400  001402        MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
4217  012260  012737  107554  001404        MOV     #BUFONE+4,RMBAO         ;CHANGE MEMORY ADDRESS
4218  012266  012737  000061  001400        MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
4219  012274  012702  001535                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
4220  012300  112722  000006                MOVB    #RMDA,(R2)+
4221  012304  112722  000034                MOVB    #RMDC,(R2)+
4222  012310  112722  000032                MOVB    #RMOF,(R2)+
4223  012314  112722  000004                MOVB    #RMBA,(R2)+
4224  012320  112722  000002                MOVB    #RMWC,(R2)+
4225  012324  112722  000000                MOVB    #RMCS1,(R2)+
4226  012330  112722  000200                MOVB    #200,(R2)+              ;TERMINATE TABLE
4227
4228  012334  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4229  012340  000404                  BR      130$                    ;GO TO 130$ IF NO ERROR
4230  012342  000240                  NOP                     ;RETURN HERE IF ERROR
4231  012344  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4232  012346  000137  013134          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4233  012352                  130$:
4234
4235                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
4236  012352  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4237
4238                          ;WAIT FOR WRITE COMMAND TO COMPLETE
4239  012356  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4240
4241                          ;GO READ STATUS FOR WRITE COMMAND
4242  012362  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4243  012366  000404                  BR      140$                    ;GO TO 140$ IF NO ERROR
4244  012370  000240                  NOP                     ;RETURN HERE IF ERROR
4245  012372  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4246  012374  000137  013134          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4247  012400                  140$:
4248
4249                          ;CHECK FOR ERRORS DURING WRITE OPERATION
4250  012400  004737  044532          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4251  012404  000405                  BR      150$                    ;GO TO 150$ IF NO ERROR
4252  012406  000240                  NOP                     ;RETURN HERE IF ERROR
4253  012410  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4254  012412  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4255  012414  000137  013134          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4256  012420                  150$:
4257  012420  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4258  012424  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
4259  012426  000240                  NOP                     ;RETURN HERE IF ERROR
```

```
4260  012430  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4261  012432  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4262  012434  000137  013134                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4263  012440                        160$:
4264
4265  012440                        170$:
4266  012440  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4267  012444  000405                        BR      180$                    ;GO TO 180$ IF NO ERROR
4268  012446  000240                        NOP                     ;RETURN HERE IF ERROR
4269  012450  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4270  012452  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4271  012454  000137  013134                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4272  012460                        180$:
4273
4274                                ;CHANGE LOOP ADDRESSES
4275  012460  012737  012502  001122        MOV     #190$,$LPADR
4276  012466  012737  012502  001124        MOV     #190$,$LPERR
4277  012474  012737  012506  001210        MOV     #195$,$ESCAPE   ;;ESCAPE TO 195$ ON ERROR
4278
4279                                ;;**************************************************************
4280                                ;LOOP #3        WRITE CHECK DATA
4281
4282  012502                        190$:
4283
4284  012502  012703  000001                MOV     #1,R3                   ;R3+WCE BIT POSITION
4285  012506  050337  110552        195$:   BIS     R3,BUFTWO-2             ;CHANGE LAST WORD OF BUFFER
4286
4287                                ;PREPARE DEVICE FOR WRITE CHECK DATA OPERATION
4288  012512  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4289  012516  054130                        .WORD   054130  ;TASK DESCRIPTOR
4290  012520  000404                        BR      200$                    ;GO TO 200$ IF NO ERROR
4291  012522  000240                        NOP                     ;RETURN HERE IF ERROR
4292  012524  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4293  012526  000137  013134                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4294  012532                        200$:
4295
4296  012532                        240$:
4297
4298                                ;READ DATA FROM DEVICE
4299  012532  012737  000051  001400        MOV     #WCD!GO,RMCS1O          ;WRITE CHECK DATA DATA COMMAND
4300  012540  012702  001535                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
4301  012544  112722  000006                MOVB    #RMDA,(R2)+
4302  012550  112722  000032                MOVB    #RMOF,(R2)+
4303  012554  112722  000034                MOVB    #RMDC,(R2)+
4304  012560  112722  000004                MOVB    #RMBA,(R2)+
4305  012564  112722  000002                MOVB    #RMWC,(R2)+
4306  012570  112722  000000                MOVB    #RMCS1,(R2)+
4307  012574  112712  000200                MOVB    #200,(R2)
4308
4309  012600  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4310  012604  000404                        BR      250$                    ;GO TO 250$ IF NO ERROR
4311  012606  000240                        NOP                     ;RETURN HERE IF ERROR
4312  012610  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4313  012612  000137  013134                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4314  012616                        250$:
4315
```

```
                                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
012616  004737  043452                      JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                        ;WAIT FOR WRITE CHECK DATA OPERATION TO COMPLETE
012622  004737  044346                      JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                        ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
012626  004737  043536                      JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
012632  000404                              BR      260$                    ;GO TO 260$ IF NO ERROR
012634  000240                              NOP                     ;RETURN HERE IF ERROR
012636  104000                              ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
012640  000137  013134                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
012644                                  260$:

                                        ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
012644  004737  044532                      JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
012650  000405                              BR      270$                    ;GO TO 270$ IF NO ERROR
012652  000240                              NOP                     ;RETURN HERE IF ERROR
012654  104000                              ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
012656  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
012660  000137  013134                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
012664                                  270$:
012664  032737  040000  001340              BIT     #WCE,RMCS2I             ;WAS "WCE" DETECTED??
012672  001030                              BNE     285$                    ;YES!!
012674  004737  057036                      JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
012700  000405                              BR      280$                    ;GO TO 280$ IF NO ERROR
012702  000240                              NOP                     ;RETURN HERE IF ERROR
012704  104000                              ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
012706  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
012710  000137  013134                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
012714                                  280$:
012714  013737  001340  001140              MOV     RMCS2I,$GDDAT           ;EXPECTED STATUS
012722  052737  040000  001140              BIS     #WCE,$GDDAT
012730  013737  001340  001142              MOV     RMCS2I,$BDDAT           ;RECEIVED STATUS
012736  010337  001174                      MOV     R3,$TMP0                ;FAILING BIT POSITION
012742  012737  110552  001176              MOV     #BUFTWO-2,$TMP1         ;FAILING ADDRESS
012750  104337                              ERROR   337                     ;WCE NOT DETECTED
012752  000470                              BR      350$

012754                                  285$:
012754  112737  000022  001506              MOVB    #RMDB,GETINX    ;SETUP GET INDEX TABLE
012762  112737  000200  001507              MOVB    #200,GETINX+1
012770  004737  043536                      JSR     PC,GET          ;GO GET THE CONTENTS OF THE DATA BUFFER
012774  000404                              BR      290$                    ;GO TO 290$ IF NO ERROR
012776  000240                              NOP                     ;RETURN HERE IF ERROR
013000  104000                              ERROR                   ;ERROR DEFINED BY GET SUBROUTINE
013002  000137  013134                      JMP     350$                    ;GO TO 350$ IF ERROR

013006                                  290$:
013006  013737  001352  001142              MOV     RMDBI,$BDDAT            ;RECEIVED DATA
013014  013737  110552  001140              MOV     BUFTWO-2,$GDDAT         ;EXPECTED DATA
013022  040337  001140                      BIC     R3,$GDDAT
013026  012737  110552  001134              MOV     #BUFTWO-2,$GDADR        ;EXPECTED ADDRESS
013034  013737  001334  001136              MOV     RMBAI,$BDADR            ;RECEIVED ADDRESS
013042  162737  000002  001136              SUB     #2,$BDADR               ;ADJUST BUS ADDRESS
013050  023737  001134  001136              CMP     $GDADR,$BDADR           ;ADDRESSES OK??
```

```
4372  013056  001402                          BEQ     295$                    ;YES!!
4373  013060  104340                          ERROR   340                     ;WCE AT UNEXPECTED ADDRESS
4374  013062  000424                          BR      350$
4375  013064  023737  001140  001142  295$:   CMP     $GDDAT,$BDDAT           ;DATA OK??
4376  013072  001402                          BEQ     296$                    ;YES!!
4377  013074  104341                          ERROR   341                     ;UNEXPECTED WCE DATA
4378  013076  000416                          BR      350$
4379  013100                          296$:
4380
4381  013100  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4382  013104  000405                          BR      300$                    ;GO TO 300$ IF NO ERROR
4383  013106  000240                          NOP                     ;RETURN HERE IF ERROR
4384  013110  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4385  013112  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4386  013114  000137  013134                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4387  013120                          300$:
4388
4389  013120  040337  110552                  BIC     R3,BUFTWO-2             ;RESTORE DATA PATTERN
4390  013124  006303                          ASL     R3                      ;SHIFT TO NEXT BIT
4391  013126  001402                          BEQ     350$                    ;EXIT IF DONE
4392  013130  000137  012506                  JMP     195$                    ;REPEAT TEST FOR NEXT DATA BIT
4393  013134                          350$:
4394  013134  012737  000000  001210          MOV     #0,$ESCAPE      ;;ESCAPE TO 0 ON ERROR
4395  013142  012737  011744  001122          MOV     #10$,$LPADR             ;CHANGE LOOP TO START OF TEST
4396  013150  012737  011744  001124          MOV     #10$,$LPERR
4397                          ;;*****************************************************************
4398                          ;*TEST 7         WRITE, READ ONES
4399                          ;*****************************************************************
4400  013156                  TST7:
4401  013156  000004                          SCOPE                   ;SCOPE CALL
4402  013160  000240                          NOP                     ;START OF TEST
4403  013162  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4404  013166  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4405  013172  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4406  013176  012737  000007  001226          MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4407
4408                          ;;*****************************************************************
4409                          ;LOOP #1        FORMAT,WRITE,READ
4410
4411  013204                  10$:
4412
4413                          ;PREPARE THE DEVICE FOR FORMAT OPERATION
4414  013204  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4415  013210  054130                          .WORD   054130  ;TASK DESCRIPTOR
4416  013212  000404                          BR      20$                     ;GO TO 20$ IF NO ERROR
4417  013214  000240                          NOP                     ;RETURN HERE IF ERROR
4418  013216  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4419  013220  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4420  013224                  20$:
4421
4422                          ;LOAD PARAMETERS AND GENERATE DATA BUFFER
4423  013224  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
4424  013232  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
4425  013240  012737  107550  001404          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
4426  013246  012737  177400  001402          MOV     #<^C256.+1>,RMWC0       ;WORD COUNT = 1 SECTOR
4427  013254  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
```

```
4428  013262  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
4429
4430                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
4431
4432  013270  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
4433  013274  000405                          BR      25$                     ;GO TO 25$ IF NO ERROR
4434  013276  104401  067466                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
4435  013302  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
4436  013304  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4437  013310                        25$:
4438  013310  012737  071046  001174          MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
4439  013316  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
4440  013324  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
4441
4442                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
4443  013330  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
4444  013334  112722  000034                  MOVB    #RMDC,(R2)+
4445  013340  112722  000006                  MOVB    #RMDA,(R2)+
4446  013344  112722  000004                  MOVB    #RMBA,(R2)+
4447  013350  112722  000002                  MOVB    #RMWC,(R2)+
4448  013354  112722  000032                  MOVB    #RMOF,(R2)+
4449  013360  112722  000000                  MOVB    #RMCS1,(R2)+
4450  013364  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
4451  013370                        30$:
4452
4453                                  ;FORMAT THE DRIVE
4454  013370  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4455  013374  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
4456  013376  000240                          NOP                     ;RETURN HERE IF ERROR
4457  013400  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4458  013402  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4459  013406                        40$:
4460
4461                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
4462  013406  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
4463
4464                                  ;WAIT FOR THE FORMAT TO COMPLETE
4465  013412  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
4466
4467                                  ;GO READ STATUS FOR FORMAT OPERATION
4468  013416  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4469  013422  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
4470  013424  000240                          NOP                     ;RETURN HERE IF ERROR
4471  013426  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4472  013430  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4473  013434                        50$:
4474
4475                                  ;VERIFY NO ERRORS DURING FORMAT
4476  013434  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
4477  013440  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
4478  013442  000240                          NOP                     ;RETURN HERE IF ERROR
4479  013444  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4480  013446  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4481  013450  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4482  013454                        60$:
4483
```

```
                                        ;MOVE LOOP ADDRESSES TO NEXT OPERATION
013454  012737  013472  001122                MOV     #70$,$LPADR
013462  012737  013472  001124                MOV     #70$,$LPERR
013470  000410                                BR      80$                     ;SKIP TO WRITE OPERATION

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

013472                                  70$:

                                        ;PREPARE DEVICE FOR WRITE OPERATION
013472  004737  040020                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
013476  054130                                .WORD   054130  ;TASK DESCRIPTOR
013500  000404                                BR      80$                     ;GO TO 80$ IF NO ERROR
013502  000240                                NOP                     ;RETURN HERE IF ERROR
013504  104000                                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
013506  000137  014200                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
013512                                  80$:

013512                                  120$:

                                        ;WRITE DATA TO THE DRIVE
013512  012737  107554  001404                MOV     #BUFONE+4,RMBAO         ;CHANGE MEMORY ADDRESS
013520  012737  177400  001402                MOV     #<-C256.>&1>,RMWCO      ;CHANGE WORD COUNT
013526  012737  000061  001400                MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
013534  012702  001535                        MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
013540  112722  000006                        MOVB    #RMDA,(R2)+
013544  112722  000034                        MOVB    #RMDC,(R2)+
013550  112722  000032                        MOVB    #RMOF,(R2)+
013554  112722  000004                        MOVB    #RMBA,(R2)+
013560  112722  000002                        MOVB    #RMWC,(R2)+
013564  112722  000000                        MOVB    #RMCS1,(R2)+
013570  112722  000200                        MOVB    #200,(R2)+              ;TERMINATE TABLE

013574  004737  044006                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
013600  000404                                BR      130$                    ;GO TO 130$ IF NO ERROR
013602  000240                                NOP                     ;RETURN HERE IF ERROR
013604  104000                                ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
013606  000137  014200                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
013612                                  130$:

                                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
013612  004737  043452                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                        ;WAIT FOR WRITE COMMAND TO COMPLETE
013616  004737  044346                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                        ;GO READ STATUS FOR WRITE COMMAND
013622  004737  043536                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
013626  000404                                BR      140$                    ;GO TO 140$ IF NO ERROR
013630  000240                                NOP                     ;RETURN HERE IF ERROR
013632  104000                                ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
013634  000137  014200                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
013640                                  140$:
```

```
4540
4541                                    ;CHECK FOR ERRORS DURING WRITE OPERATION
4542  013640  004737  044532                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4543  013644  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
4544  013646  000240                            NOP                     ;RETURN HERE IF ERROR
4545  013650  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4546  013652  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4547  013654  000137  014200                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4548  013660                            150$:
4549  013660  004737  057036                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4550  013664  000405                            BR      160$                    ;GO TO 160$ IF NO ERROR
4551  013666  000240                            NOP                     ;RETURN HERE IF ERROR
4552  013670  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4553  013672  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4554  013674  000137  014200                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4555  013700                            160$:
4556
4557  013700                            170$:
4558  013700  004737  045364                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4559  013704  000405                            BR      180$                    ;GO TO 180$ IF NO ERROR
4560  013706  000240                            NOP                     ;RETURN HERE IF ERROR
4561  013710  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4562  013712  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4563  013714  000137  014200                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4564  013720                            180$:
4565
4566                                    ;CHANGE LOOP ADDRESSES
4567  013720  012737  013736  001122            MOV     #190$,$LPADR
4568  013726  012737  013736  001124            MOV     #190$,$LPERR
4569  013734  000410                            BR      200$                    ;SKIP TO NEXT OPERATION
4570
4571                                    ;;*****************************************************************
4572                                    ;LOOP #3        READ
4573
4574  013736                            190$:
4575
4576                                    ;PREPARE DEVICE FOR READ OPERATION
4577  013736  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4578  013742  054130                            .WORD   054130  ;TASK DESCRIPTOR
4579  013744  000404                            BR      200$                    ;GO TO 200$ IF NO ERROR
4580  013746  000240                            NOP                     ;RETURN HERE IF ERROR
4581  013750  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4582  013752  000137  014200                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4583  013756                            200$:
4584
4585  013756                            240$:
4586
4587                                    ;READ DATA FROM DEVICE
4588  013756  012737  110560  001404            MOV     #BUFTWO+4,RMBA0         ;CHANGE MEMORY ADDRESS
4589  013764  012737  000071  001400            MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
4590  013772  012702  001535                    MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
4591  013776  112722  000006                    MOVB    #RMDA,(R2)+
4592  014002  112722  000032                    MOVB    #RMOF,(R2)+
4593  014006  112722  000034                    MOVB    #RMDC,(R2)+
4594  014012  112722  000004                    MOVB    #RMBA,(R2)+
4595  014016  112722  000002                    MOVB    #RMWC,(R2)+
```

```
4596  014022  112722  000000                  MOVB    #RMCS1,(R2)+
4597  014026  112712  000200                  MOVB    #200,(R2)
4598
4599  014032  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4600  014036  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
4601  014040  000240                          NOP                     ;RETURN HERE IF ERROR
4602  014042  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4603  014044  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4604  014050                          250$:
4605
4606                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
4607  014050  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4608
4609                                  ;WAIT FOR READ OPERATION TO COMPLETE
4610  014054  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4611
4612                                  ;GO READ STATUS FOR READ OPERATION
4613  014060  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4614  014064  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
4615  014066  000240                          NOP                     ;RETURN HERE IF ERROR
4616  014070  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4617  014072  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4618  014076                          260$:
4619
4620                                  ;CHECK FOR ERRORS DURING READ OPERATION
4621  014076  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4622  014102  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
4623  014104  000240                          NOP                     ;RETURN HERE IF ERROR
4624  014106  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4625  014110  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4626  014112  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4627  014116                          270$:
4628  014116  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4629  014122  000405                          BR      280$                    ;GO TO 280$ IF NO ERROR
4630  014124  000240                          NOP                     ;RETURN HERE IF ERROR
4631  014126  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4632  014130  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4633  014132  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4634  014136                          280$:
4635
4636  014136                          290$:
4637  014136  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4638  014142  000405                          BR      300$                    ;GO TO 300$ IF NO ERROR
4639  014144  000240                          NOP                     ;RETURN HERE IF ERROR
4640  014146  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4641  014150  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4642  014152  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4643  014156                          300$:
4644  014156  004737  043104                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
4645  014162  107554                          .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
4646  014164  110560                          .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
4647  014166  000404                          BR      310$                    ;GO TO 310$ IF NO ERROR
4648  014170  000240                          NOP                     ;RETURN HERE IF ERROR
4649  014172  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
4650  014174  000137  014200                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4651  014200                          310$:
```

```
                                ;;*****************************************************************
                                ;*TEST 10        WRITE, WRITE CHECK ONES
                                ;;*****************************************************************
014200                          TST10:
014200  000004                          SCOPE                   ;SCOPE CALL
014202  000240                          NOP                     ;START OF TEST
014204  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
014210  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
014214  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
014220  012737  000010  001226          MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                                ;;*****************************************************************
                                ;LOOP #1        FORMAT,WRITE,WRITE CHECK DATA

014226                          10$:

                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
014226  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
014232  054130                          .WORD   054130  ;TASK DESCRIPTOR
014234  000404                          BR      20$             ;GO TO 20$ IF NO ERROR
014236  000240                          NOP                     ;RETURN HERE IF ERROR
014240  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
014242  000137  015172                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
014246                          20$:

                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
014246  012737  000000  001434          MOV     #0,RMDC0        ;CYLINDER = 0
014254  012737  000000  001406          MOV     #0,RMDA0        ;TRACK = 0, SECTOR = 0
014262  012737  107550  001404          MOV     #BUFONE,RMBA0   ;BUS ADDRESS
014270  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT = 1 SECTOR
014276  012737  010000  001432          MOV     #FMT16,RM0F0    ;16 BIT FORMAT
014304  012737  000063  001400          MOV     #WH!GO,RMCS10   ;WRITE HEADER AND DATA COMMAND

                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
014312  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
014316  000405                          BR      25$             ;GO TO 25$ IF NO ERROR
014320  104401  067466                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
014324  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
014326  000137  015172                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
014332                          25$:
014332  012737  071046  001174          MOV     #ONES,$TMP0     ;STARTING ADDRESS OF PATTERN
014340  012737  000001  001176          MOV     #1,$TMP1        ;RANGE OF PATTERN
014346  004737  042640                  JSR     PC,GENBUF       ;GO GENERATE BUFFER FOR FORMAT

                                ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
014352  012702  001535                  MOV     #PUTINX,R2      ;R2 = BYTE ENTRY POSITION
014356  112722  000034                  MOVB    #RMDC,(R2)+
014362  112722  000006                  MOVB    #RMDA,(R2)+
014366  112722  000004                  MOVB    #RMBA,(R2)+
014372  112722  000002                  MOVB    #RMWC,(R2)+
014376  112722  000032                  MOVB    #RMOF,(R2)+
014402  112722  000000                  MOVB    #RMCS1,(R2)+
014406  112712  000200                  MOVB    #200,(R2)       ;TERMINATE TABLE
```

```
4708  014412                        30$:
4709
4710                                ;FORMAT THE DRIVE
4711  014412  004737  044006              JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4712  014416  000404                      BR      40$                     ;GO TO 40$ IF NO ERROR
4713  014420  000240                      NOP                     ;RETURN HERE IF ERROR
4714  014422  104000                      ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4715  014424  000137  015172              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4716  014430                        40$:
4717
4718                                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
4719  014430  004737  043452              JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
4720
4721                                ;WAIT FOR THE FORMAT TO COMPLETE
4722  014434  004737  044346              JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
4723
4724                                ;GO READ STATUS FOR FORMAT OPERATION
4725  014440  004737  043536              JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4726  014444  000404                      BR      50$                     ;GO TO 50$ IF NO ERROR
4727  014446  000240                      NOP                     ;RETURN HERE IF ERROR
4728  014450  104000                      ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4729  014452  000137  015172              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4730  014456                        50$:
4731
4732                                ;VERIFY NO ERRORS DURING FORMAT
4733  014456  004737  057036              JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
4734  014462  000405                      BR      60$                     ;GO TO 60$ IF NO ERROR
4735  014464  000240                      NOP                     ;RETURN HERE IF ERROR
4736  014466  104000                      ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4737  014470  004736                      JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4738  014472  000137  015172              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4739  014476                        60$:
4740
4741                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
4742  014476  012737  014514  001122        MOV     #70$,$LPADR
4743  014504  012737  014514  001124        MOV     #70$,$LPERR
4744  014512  000410                      BR      80$                             ;SKIP TO WRITE OPERATION
4745
4746                                ;;*****************************************************************
4747                                ;LOOP #2        WRITE,WRITE CHECK DATA
4748
4749  014514                        70$:
4750
4751
4752                                ;PREPARE DEVICE FOR WRITE OPERATION
4753  014514  004737  040020              JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
4754  014520  054130                      .WORD   054130  ;TASK DESCRIPTOR
4755  014522  000404                      BR      80$                     ;GO TO 80$ IF NO ERROR
4756  014524  000240                      NOP                     ;RETURN HERE IF ERROR
4757  014526  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4758  014530  000137  015172              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4759  014534                        80$:
4760
4761
4762  014534                        120$:
4763
```

```
4764
4765  014534  012737  107554  001404          ;WRITE DATA TO THE DRIVE
4766  014542  012737  177400  001402                MOV     #BUFONE+4,RMBA0         ;CHANGE MEMORY ADDRESS
4767  014550  012737  000061  001400                MOV     #<^C256,+1>,RMWC0       ;CHANGE WORD COUNT
4768  014556  012702  001535                      MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
4769  014562  112722  000006                      MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
4770  014566  112722  000034                      MOVB    #RMDA,(R2)+
4771  014572  112722  000032                      MOVB    #RMDC,(R2)+
4772  014576  112722  000004                      MOVB    #RMOF,(R2)+
4773  014602  112722  000002                      MOVB    #RMBA,(R2)+
4774  014606  112722  000000                      MOVB    #RMWC,(R2)+
4775  014612  112722  000200                      MOVB    #RMCS1,(R2)+
4776                                              MOVB    #200,(R2)+              ;TERMINATE TABLE
4777  014616  004737  044006                      JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4778  014622  000404                              BR      130$                    ;GO TO 130$ IF NO ERROR
4779  014624  000240                              NOP                     ;RETURN HERE IF ERROR
4780  014626  104000                              ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4781  014630  000137  015172                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4782  014634                          130$:
4783
4784                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
4785  014634  004737  043452                      JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4786
4787                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
4788  014640  004737  044346                      JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4789
4790                                  ;GO READ STATUS FOR WRITE COMMAND
4791  014644  004737  043536                      JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4792  014650  000404                              BR      140$                    ;GO TO 140$ IF NO ERROR
4793  014652  000240                              NOP                     ;RETURN HERE IF ERROR
4794  014654  104000                              ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4795  014656  000137  015172                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4796  014662                          140$:
4797
4798                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
4799  014662  004737  044532                      JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4800  014666  000405                              BR      150$                    ;GO TO 150$ IF NO ERROR
4801  014670  000240                              NOP                     ;RETURN HERE IF ERROR
4802  014672  104000                              ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4803  014674  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4804  014676  000137  015172                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4805  014702                          150$:
4806  014702  004737  057036                      JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4807  014706  000405                              BR      160$                    ;GO TO 160$ IF NO ERROR
4808  014710  000240                              NOP                     ;RETURN HERE IF ERROR
4809  014712  104000                              ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4810  014714  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4811  014716  000137  015172                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4812  014722                          160$:
4813
4814  014722                          170$:
4815  014722  004737  045364                      JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4816  014726  000405                              BR      180$                    ;GO TO 180$ IF NO ERROR
4817  014730  000240                              NOP                     ;RETURN HERE IF ERROR
4818  014732  104000                              ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4819  014734  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
4820  014736  000137  015172                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4821  014742                          180$:
4822
4823                                  ;CHANGE LOOP ADDRESSES
4824  014742  012737  014760  001122          MOV     #190$,$LPADR
4825  014750  012737  014760  001124          MOV     #190$,$LPERR
4826  014756  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
4827
4828                                  ;;*****************************************************************
4829                                  ;LOOP #3        WRITE CHECK DATA
4830
4831  014760                          190$:
4832
4833                                  ;PREPARE DEVICE FOR WRITE CHECK DATA OPERATION
4834  014760  004737  040020                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
4835  014764  054130                          .WORD   054130  ;TASK DESCRIPTOR
4836  014766  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
4837  014770  000240                          NOP                             ;RETURN HERE IF ERROR
4838  014772  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4839  014774  000137  015172                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4840  015000                          200$:
4841
4842  015000                          240$:
4843
4844                                  ;WRITE CHECK DATA DATA FROM DEVICE
4845  015000  012737  000051  001400          MOV     #WCD!GO,RMCS10          ;WRITE CHECK DATA DATA COMMAND
4846  015006  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
4847  015012  112722  000006                  MOVB    #RMDA,(R2)+
4848  015016  112722  000032                  MOVB    #RMOF,(R2)+
4849  015022  112722  000034                  MOVB    #RMDC,(R2)+
4850  015026  112722  000004                  MOVB    #RMBA,(R2)+
4851  015032  112722  000002                  MOVB    #RMWC,(R2)+
4852  015036  112722  000000                  MOVB    #RMCS1,(R2)+
4853  015042  112712  000200                  MOVB    #200,(R2)
4854
4855  015046  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4856  015052  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
4857  015054  000240                          NOP                             ;RETURN HERE IF ERROR
4858  015056  104000                          ERROR                           ;ERROR # DEFINED BY PUT SUBROUTINE
4859  015060  000137  015172                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4860  015064                          250$:
4861
4862                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
4863  015064  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
4864
4865                                  ;WAIT FOR WRITE CHECK DATA OPERATION TO COMPLETE
4866  015070  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
4867
4868                                  ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
4869  015074  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4870  015100  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
4871  015102  000240                          NOP                             ;RETURN HERE IF ERROR
4872  015104  104000                          ERROR                           ;ERROR # DEFINED BY GET SUBROUTINE
4873  015106  000137  015172                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4874  015112                          260$:
4875
```

```
                                ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
015112  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
015116  000405                      BR      270$                    ;GO TO 270$ IF NO ERROR
015120  000240                      NOP                     ;RETURN HERE IF ERROR
015122  104000                      ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
015124  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
015126  000137  015172              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
015132                          270$:
015132  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
015136  000405                      BR      280$                    ;GO TO 280$ IF NO ERROR
015140  000240                      NOP                     ;RETURN HERE IF ERROR
015142  104000                      ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
015144  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
015146  000137  015172              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
015152                          280$:
                                290$:
015152  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
015156  000405                      BR      300$                    ;GO TO 300$ IF NO ERROR
015160  000240                      NOP                     ;RETURN HERE IF ERROR
015162  104000                      ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
015164  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
015166  000137  015172              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
015172                          300$:
                                350$:
015172                          ;;*****************************************************************
                                ;*TEST 11        WRITE, WRITE CHECK ONES W/ WCE ERROR
                                ;;*****************************************************************
015172                          TST11:
015172  000004                      SCOPE                   ;SCOPE CALL
015174  000240                      NOP                     ;START OF TEST
015176  012706  001100              MOV     #STACK,SP       ;INITIALIZE STACK POINTER
015202  013700  001276              MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
015206  013701  001450              MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
015212  012737  000011  001226      MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                                ;;*****************************************************************
                                ;LOOP #1        FORMAT,WRITE,WRITE CHECK DATA

015220                          10$:

                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
015220  004737  040020              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
015224  054130                      .WORD   054130  ;TASK DESCRIPTOR
015226  000404                      BR      20$                     ;GO TO 20$ IF NO ERROR
015230  000240                      NOP                     ;RETURN HERE IF ERROR
015232  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
015234  016412  016412              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
015240                          20$:

                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
015240  012737  000000  001434      MOV     #0,RMDC0                ;CYLINDER = 0
015246  012737  000000  001406      MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
015254  012737  107550  001404      MOV     #BUFONE,RMBA0           ;BUS ADDRESS
015262  012737  177376  001402      MOV     #^C<2+256.>+1>,RMWC0            ;WORD COUNT = 1 SECTOR
```

```
4932  015270  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
4933  015276  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
4934
4935                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
4936
4937  015304  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
4938  015310  000405                          BR      25$                     ;GO TO 25$ IF NO ERROR
4939  015312  104401  067466                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
4940  015316  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
4941  015320  000137  016412                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4942  015324                          25$:
4943  015324  012737  071046  001174          MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
4944  015332  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
4945  015340  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
4946
4947                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
4948  015344  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
4949  015350  112722  000034                  MOVB    #RMDC,(R2)+
4950  015354  112722  000006                  MOVB    #RMDA,(R2)+
4951  015360  112722  000004                  MOVB    #RMBA,(R2)+
4952  015364  112722  000002                  MOVB    #RMWC,(R2)+
4953  015370  112722  000032                  MOVB    #RMOF,(R2)+
4954  015374  112722  000000                  MOVB    #RMCS1,(R2)+
4955  015400  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
4956  015404                          30$:
4957
4958                                  ;FORMAT THE DRIVE
4959  015404  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
4960  015410  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
4961  015412  000240                          NOP                     ;RETURN HERE IF ERROR
4962  015414  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
4963  015416  000137  016412                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4964  015422                          40$:
4965
4966                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
4967  015422  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
4968
4969                                  ;WAIT FOR THE FORMAT TO COMPLETE
4970  015426  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
4971
4972                                  ;GO READ STATUS FOR FORMAT OPERATION
4973  015432  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
4974  015436  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
4975  015440  000240                          NOP                     ;RETURN HERE IF ERROR
4976  015442  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
4977  015444  000137  016412                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4978  015450                          50$:
4979
4980                                  ;VERIFY NO ERRORS DURING FORMAT
4981  015450  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4982  015454  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
4983  015456  000240                          NOP                     ;RETURN HERE IF ERROR
4984  015460  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4985  015462  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4986  015464  000137  016412                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
4987  015470                          60$:
```

```
                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
015470  012737  015506  001122          MOV     #70$,$LPADR
015476  012737  015506  001124          MOV     #70$,$LPERR
015504  000410                          BR      80$                     ;SKIP TO WRITE OPERATION

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

015506                          70$:

                                ;PREPARE DEVICE FOR WRITE OPERATION
015506  004737  040020                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
015512  054130                          .WORD   054130  ;TASK DESCRIPTOR
015514  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
015516  000240                          NOP                             ;RETURN HERE IF ERROR
015520  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
015522  000137  016412                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
015526                          80$:

015526                          120$:

                                ;WRITE DATA TO THE DRIVE
015526  012737  177400  001402          MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
015534  012737  107554  001404          MOV     #BUFONE+4,RMBAO         ;CHANGE MEMORY ADDRESS
015542  012737  000061  001400          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
015550  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
015554  112722  000006                  MOVB    #RMDA,(R2)+
015560  112722  000034                  MOVB    #RMDC,(R2)+
015564  112722  000032                  MOVB    #RMOF,(R2)+
015570  112722  000004                  MOVB    #RMBA,(R2)+
015574  112722  000002                  MOVB    #RMWC,(R2)+
015600  112722  000000                  MOVB    #RMCS1,(R2)+
015604  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE

015610  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
015614  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
015616  000240                          NOP                             ;RETURN HERE IF ERROR
015620  104000                          ERROR                           ;ERROR # DEFINED BY PUT SUBROUTINE
015622  000137  016412                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
015626                          130$:

                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
015626  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR WRITE COMMAND TO COMPLETE
015632  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR WRITE COMMAND
015636  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
015642  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
015644  000240                          NOP                             ;RETURN HERE IF ERROR
015646  104000                          ERROR                           ;ERROR # DEFINED BY GET SUBROUTINE
015650  000137  016412                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
```

```
5044  015654                                  140$:
5045
5046                                          ;CHECK FOR ERRORS DURING WRITE OPERATION
5047  015654  004737  044532                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5048  015660  000405                                BR      150$                    ;GO TO 150$ IF NO ERROR
5049  015662  000240                                NOP                     ;RETURN HERE IF ERROR
5050  015664  104000                                ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5051  015666  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5052  015670  000137  016412                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5053  015674                                  150$:
5054  015674  004737  057036                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5055  015700  000405                                BR      160$                    ;GO TO 160$ IF NO ERROR
5056  015702  000240                                NOP                     ;RETURN HERE IF ERROR
5057  015704  104000                                ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5058  015706  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5059  015710  000137  016412                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5060  015714                                  160$:
5061
5062  015714                                  170$:
5063  015714  004737  045364                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5064  015720  000405                                BR      180$                    ;GO TO 180$ IF NO ERROR
5065  015722  000240                                NOP                     ;RETURN HERE IF ERROR
5066  015724  104000                                ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5067  015726  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5068  015730  000137  016412                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5069  015734                                  180$:
5070
5071                                          ;CHANGE LOOP ADDRESSES
5072  015734  012737  015756  001122                MOV     #190$,$LPADR
5073  015742  012737  015756  001124                MOV     #190$,$LPERR
5074  015750  012737  015762  001210                MOV     #195$,$ESCAPE   ;;ESCAPE TO 195$ ON ERROR
5075
5076                                          ;;*****************************************************************
5077                                          ;LOOP #3        WRITE CHECK DATA
5078
5079  015756                                  190$:
5080
5081  015756  012703  000001                        MOV     #1,R3                   ;R3+WCE BIT POSITION
5082  015762  040337  110552                195$:   BIC     R3,BUFTWO-2             ;CHANGE LAST WORD OF BUFFER
5083
5084                                          ;PREPARE DEVICE FOR WRITE CHECK DATA OPERATION
5085  015766  004737  040020                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5086  015772  054130                                .WORD   054130  ;TASK DESCRIPTOR
5087  015774  000404                                BR      200$                    ;GO TO 200$ IF NO ERROR
5088  015776  000240                                NOP                     ;RETURN HERE IF ERROR
5089  016000  104000                                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5090  016002  000137  016412                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5091  016006                                  200$:
5092
5093  016006                                  240$:
5094
5095
5096  016006  012737  000051  001400                MOV     #WCD!GO,RMCS10          ;WRITE CHECK DATA DATA COMMAND
5097  016014  012702  001535                        MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5098  016020  112722  000006                        MOVB    #RMDA,(R2)+
5099  016024  112722  000032                        MOVB    #RMOF,(R2)+
```

```
5100  016030  112722  000034                    MOVB    #RMDC,(R2)+
5101  016034  112722  000004                    MOVB    #RMBA,(R2)+
5102  016040  112722  000002                    MOVB    #RMWC,(R2)+
5103  016044  112722  000000                    MOVB    #RMCS1,(R2)+
5104  016050  112712  000200                    MOVB    #200,(R2)
5105
5106  016054  004737  044006                    JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5107  016060  000404                            BR      250$                    ;GO TO 250$ IF NO ERROR
5108  016062  000240                            NOP                     ;RETURN HERE IF ERROR
5109  016064  104000                            ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5110  016066  000137  016412                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5111  016072                          250$:
5112
5113                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5114  016072  004737  043452                    JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5115
5116                                  ;WAIT FOR WRITE CHECK DATA OPERATION TO COMPLETE
5117  016076  004737  044346                    JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5118
5119                                  ;GO READ STATUS FOR WRITE CHECK DATA OPERATION
5120  016102  004737  043536                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5121  016106  000404                            BR      260$                    ;GO TO 260$ IF NO ERROR
5122  016110  000240                            NOP                     ;RETURN HERE IF ERROR
5123  016112  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5124  016114  000137  016412                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5125  016120                          260$:
5126
5127                                  ;CHECK FOR ERRORS DURING WRITE CHECK DATA OPERATION
5128  016120  004737  044532                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5129  016124  000405                            BR      270$                    ;GO TO 270$ IF NO ERROR
5130  016126  000240                            NOP                     ;RETURN HERE IF ERROR
5131  016130  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5132  016132  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5133  016134  000137  016412                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5134  016140                          270$:
5135  016140  032737  040000  001340            BIT     #WCE,RMCS2I             ;WAS "WCE" DETECTED??
5136  016146  001030                            BNE     285$                    ;YES!!
5137  016150  004737  057036                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5138  016154  000405                            BR      280$                    ;GO TO 280$ IF NO ERROR
5139  016156  000240                            NOP                     ;RETURN HERE IF ERROR
5140  016160  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5141  016162  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5142  016164  000137  016412                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5143  016170                          280$:
5144  016170  013737  001340  001140            MOV     RMCS2I,$GDDAT           ;EXPECTED STATUS
5145  016176  052737  040000  001140            BIS     #WCE,$GDDAT
5146  016204  013737  001340  001142            MOV     RMCS2I,$BDDAT           ;RECEIVED STATUS
5147  016212  010337  001174                    MOV     R3,$TMP0                ;FAILING BIT POSITION
5148  016216  012737  110552  001176            MOV     #BUFTWO-2,$TMP1         ;FAILING ADDRESS
5149  016224  104337                            ERROR   337                     ;WCE NOT DETECTED
5150  016226  000471                            BR      350$
5151
5152  016230                          285$:
5153  016230  112737  000022  001506            MOVB    #RMDB,GETINX    ;SETUP GET INDEX TABLE
5154  016236  112737  000200  001507            MOVB    #200,GETINX+1
5155  016244  012737  016412  001352            MOV     #350$,RMDBI     ;SET THE INPUT BUFFER
```

```
5156  016252  004737  043536          JSR     PC,GET          ;GO GET THE CONTENTS OF THE DATA BUFFER
5157  016256  000402                  BR      290$            ;GO TO 290$ IF NO ERROR
5158  016260  000240                  NOP                     ;RETURN HERE IF ERROR
5159  016262  104000                  ERROR                   ;ERROR DEFINED BY GET SUBROUTINE
5160
5161  016264                  290$:
5162  016264  013737  001352  001142  MOV     RMDBI,$BDDAT            ;RECEIVED DATA
5163  016272  013737  110552  001140  MOV     BUFTWO-2,$GDDAT         ;EXPECTED DATA
5164  016300  050337  001140          BIS     R3,$GDDAT
5165  016304  012737  110552  001134  MOV     #BUFTWO-2,$GDADR        ;EXPECTED ADDRESS
5166  016312  013737  001334  001136  MOV     RMBAI,$BDADR            ;RECEIVED ADDRESS
5167  016320  162737  000002  001136  SUB     #2,$BDADR               ;CORRECT MEMORY ADDRESS
5168  016326  023737  001134  001136  CMP     $GDADR,$BDADR           ;ADDRESSES OK??
5169  016334  001402                  BEQ     295$                    ;YES!!
5170  016336  104340                  ERROR   340                     ;WCE AT UNEXPECTED ADDRESS
5171  016340  000424                  BR      350$
5172  016342  023737  001140  001142  295$:   CMP     $GDDAT,$BDDAT           ;DATA OK??
5173  016350  001402                  BEQ     296$                    ;YES!!
5174  016352  104341                  ERROR   341                     ;UNEXPECTED WCE DATA
5175  016354  000416                  BR      350$
5176  016356                  296$:
5177
5178  016356  004737  045364          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5179  016362  000405                  BR      300$                    ;GO TO 300$ IF NO ERROR
5180  016364  000240                  NOP                     ;RETURN HERE IF ERROR
5181  016366  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5182  016370  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5183  016372  000137  016412          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5184  016376                  300$:
5185
5186  016376  050337  110552          BIS     R3,BUFTWO-2             ;RESTORE DATA PATTERN
5187  016402  006303                  ASL     R3                      ;SHIFT TO NEXT BIT
5188  016404  001402                  BEQ     350$                    ;EXIT IF DONE
5189  016406  000137  015762          JMP     195$                    ;REPEAT TEST FOR NEXT DATA BIT
5190  016412                  350$:
5191  016412  012737  000000  001210  MOV     #0,$ESCAPE      ;;ESCAPE TO 0 ON ERROR
5192  016420  012737  015220  001122  MOV     #10$,$LPADR             ;CHANGE LOOP TO START OF TEST
5193  016426  012737  015220  001124  MOV     #10$,$LPERR
5194                  ;;**************************************************************
5195                  ;*TEST 12        WRITE, WRITE CHECK MULTIPLE SECTORS
5196                  ;;**************************************************************
5197  016434          ;ST12:
5198  016434  000004                          SCOPE                   ;SCOPE CALL
5199  016436  000240                          NOP                     ;START OF TEST
5200  016440  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5201  016444  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5202  016450  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5203  016454  012737  000012  001226          MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5204
5205                  ;;**************************************************************
5206                  ;LOOP #1        FORMAT,WRITE,READ
5207
5208  016462          10$:
5209
5210                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
5211  016462  004737  040020                  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*<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      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
                                25$:
                                        MOV     #ZEROS,$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
```

```
5268  016704  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5269  016706  000137  017454                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5270  016712                        50$:
5271
5272                                ;VERIFY NO ERRORS DURING FORMAT
5273  016712  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5274  016716  000405                        BR      60$                     ;GO TO 60$ IF NO ERROR
5275  016720  000240                        NOP                     ;RETURN HERE IF ERROR
5276  016722  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5277  016724  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5278  016726  000137  017454                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5279  016732                        60$:
5280
5281                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
5282  016732  012737  017012  001122        MOV     #70$,$LPADR
5283  016740  012737  017012  001124        MOV     #70$,$LPERR
5284                                ;REGENERATE DATA BUFFER
5285  016746  012737  177000  001402        MOV     #<^C<2*256.>+1>,RMWCO   ;CHANGE WORD COUNT
5286  016754  012737  107550  001404        MOV     #BUFONE,RMBAO           ;CHANGE BUS ADDRESS
5287  016762  012737  000060  001400        MOV     #WD,RMCS1O              ;WRITE DATA
5288  016770  012737  071046  001174        MOV     #ONES,$TMPO             ;STARTING ADDRESS OF PATTERN
5289  016776  012737  000001  001176        MOV     #1,$TMP1                ;RANGE OF PATTERN
5290  017004  004737  042640                JSR     PC,GENBUF
5291  017010  000410                        BR      80$                     ;SKIP TO WRITE OPERATION
5292
5293                                ;;*****************************************************************
5294                                ;LOOP #2        WRITE,READ
5295
5296  017012                        70$:
5297
5298
5299                                ;PREPARE DEVICE FOR WRITE OPERATION
5300  017012  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5301  017016  054130                        .WORD   054130  ;TASK DESCRIPTOR
5302  017020  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
5303  017022  000240                        NOP                     ;RETURN HERE IF ERROR
5304  017024  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5305  017026  000137  017454                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5306  017032                        80$:
5307
5308
5309  017032                        120$:
5310
5311                                ;WRITE DATA TO THE DRIVE
5312  017032  012737  000061  001400        MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
5313  017040  012702  001535                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5314  017044  112722  000006                MOVB    #RMDA,(R2)+
5315  017050  112722  000034                MOVB    #RMDC,(R2)+
5316  017054  112722  000032                MOVB    #RMOF,(R2)+
5317  017060  112722  000004                MOVB    #RMBA,(R2)+
5318  017064  112722  000002                MOVB    #RMWC,(R2)+
5319  017070  112722  000000                MOVB    #RMCS1,(R2)+
5320  017074  112722  000200                MOVB    #200,(R2)+              ;TERMINATE TABLE
5321
5322  017100  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5323  017104  000404                        BR      130$                    ;GO TO 130$ IF NO ERROR
```

```
5324  017106  000240                          NOP                     ;RETURN HERE IF ERROR
5325  017110  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5326  017112  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5327  017116                          130$:
5328
5329                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5330  017116  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5331
5332                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
5333  017122  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5334
5335                                  ;GO READ STATUS FOR WRITE COMMAND
5336  017126  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5337  017132  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
5338  017134  000240                          NOP                     ;RETURN HERE IF ERROR
5339  017136  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5340  017140  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5341  017144                          140$:
5342
5343                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
5344  017144  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5345  017150  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
5346  017152  000240                          NOP                     ;RETURN HERE IF ERROR
5347  017154  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5348  017156  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5349  017160  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5350  017164                          150$:
5351  017164  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5352  017170  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
5353  017172  000240                          NOP                     ;RETURN HERE IF ERROR
5354  017174  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5355  017176  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5356  017200  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5357  017204                          160$:
5358
5359  017204                          170$:
5360  017204  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5361  017210  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
5362  017212  000240                          NOP                     ;RETURN HERE IF ERROR
5363  017214  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5364  017216  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5365  017220  000137  017454                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5366  017224                          180$:
5367
5368                                  ;CHANGE LOOP ADDRESSES
5369  017224  012737  017242  001122          MOV     #190$,$LPADR
5370  017232  012737  017242  001124          MOV     #190$,$LPERR
5371  017240  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
5372
5373                                  ;;*****************************************************************
5374                                  ;LOOP #3        READ
5375
5376  017242                          190$:
5377
5378                                  ;PREPARE DEVICE FOR READ OPERATION
5379  017242  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
```

```
5380  017246  054130                        .WORD   054130  ;TASK DESCRIPTOR
5381  017250  000404                        BR      200$                    ;GO TO 200$ IF NO ERROR
5382  017252  000240                        NOP                     ;RETURN HERE IF ERROR
5383  017254  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5384  017256  000137  017454                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5385  017262                        200$:
5386
5387  017262                        240$:
5388
5389                                ;READ DATA FROM DEVICE
5390  017262  012737  000051  001400        MOV     #WCD!GO,RMCS10          ;READ DATA COMMAND
5391  017270  012702  001535                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5392  017274  112722  000006                MOVB    #RMDA,(R2)+
5393  017300  112722  000032                MOVB    #RMOF,(R2)+
5394  017304  112722  000034                MOVB    #RMDC,(R2)+
5395  017310  112722  000004                MOVB    #RMBA,(R2)+
5396  017314  112722  000002                MOVB    #RMWC,(R2)+
5397  017320  112722  000000                MOVB    #RMCS1,(R2)+
5398  017324  112712  000200                MOVB    #200,(R2)
5399
5400  017330  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5401  017334  000404                        BR      250$                    ;GO TO 250$ IF NO ERROR
5402  017336  000240                        NOP                     ;RETURN HERE IF ERROR
5403  017340  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5404  017342  000137  017454                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5405  017346                        250$:
5406
5407                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5408  017346  004737  043452                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5409
5410                                ;WAIT FOR READ OPERATION TO COMPLETE
5411  017352  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5412
5413                                ;GO READ STATUS FOR READ OPERATION
5414  017356  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5415  017362  000404                        BR      260$                    ;GO TO 260$ IF NO ERROR
5416  017364  000240                        NOP                     ;RETURN HERE IF ERROR
5417  017366  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5418  017370  000137  017454                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5419  017374                        260$:
5420
5421                                ;CHECK FOR ERRORS DURING READ OPERATION
5422  017374  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5423  017400  000405                        BR      270$                    ;GO TO 270$ IF NO ERROR
5424  017402  000240                        NOP                     ;RETURN HERE IF ERROR
5425  017404  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5426  017406  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5427  017410  000137  017454                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5428  017414                        270$:
5429  017414  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5430  017420  000405                        BR      280$                    ;GO TO 280$ IF NO ERROR
5431  017422  000240                        NOP                     ;RETURN HERE IF ERROR
5432  017424  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5433  017426  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5434  017430  000137  017454                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5435  017434                        280$:
```

```
5436
5437  017434                          290$:
5438  017434  004737  045364          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5439  017440  000405                  BR      300$                    ;GO TO 300$ IF NO ERROR
5440  017442  000240                  NOP                     ;RETURN HERE IF ERROR
5441  017444  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5442  017446  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5443  017450  000137  017454          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5444  017454                          300$:
5445
5446  017454                          350$:
5447                                  ;;*****************************************************************
5448                                  ;*TEST 13        WRITE, READ W/ IMPLIED SEEK
5449                                  ;;*****************************************************************
5450  017454                          TST13:
5451  017454  000004                  SCOPE                   ;SCOPE CALL
5452  017456  000240                  NOP                     ;START OF TEST
5453  017460  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5454  017464  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5455  017470  013701  001450          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5456  017474  012737  000013  001226  MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5457
5458                                  ;;*****************************************************************
5459                                  ;LOOP #1        FORMAT,WRITE,READ
5460
5461  017502                          10$:
5462
5463                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
5464  017502  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5465  017506  054130                  .WORD   054130  ;TASK DESCRIPTOR
5466  017510  000404                  BR      20$                     ;GO TO 20$ IF NO ERROR
5467  017512  000240                  NOP                     ;RETURN HERE IF ERROR
5468  017514  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5469  017516  000137  020776          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5470  017522                          20$:
5471
5472                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
5473  017522  012737  000000  001434  MOV     #0,RMDC0                ;CYLINDER = 0
5474  017530  012737  000000  001406  MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
5475  017536  012737  107550  001404  MOV     #BUFONE,RMBA0           ;BUS ADDRESS
5476  017544  012737  177376  001402  MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
5477  017552  012737  010000  001432  MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5478  017560  012737  000063  001400  MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
5479
5480                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
5481
5482  017566  004737  040734          JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
5483  017572  000405                  BR      25$                     ;GO TO 25$ IF NO ERROR
5484  017574  104401  067466          TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
5485  017600  104000                  ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
5486  017602  000137  020776          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5487  017606                          25$:
5488  017606  012737  071046  001174  MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
5489  017614  012737  000001  001176  MOV     #1,$TMP1                ;RANGE OF PATTERN
5490  017622  004737  042640          JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
5491
```

```
                                ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
017626  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
017632  112722  000034                  MOVB    #RMDC,(R2)+
017636  112722  000006                  MOVB    #RMDA,(R2)+
017642  112722  000004                  MOVB    #RMBA,(R2)+
017646  112722  000002                  MOVB    #RMWC,(R2)+
017652  112722  000032                  MOVB    #RMOF,(R2)+
017656  112722  000000                  MOVB    #RMCS1,(R2)+
017662  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
017666                          30$:

                                ;FORMAT THE DRIVE
017666  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
017672  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
017674  000240                          NOP                     ;RETURN HERE IF ERROR
017676  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
017700  000137  020776                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
017704                          40$:

                                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
017704  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR THE FORMAT TO COMPLETE
017710  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR FORMAT OPERATION
017714  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
017720  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
017722  000240                          NOP                     ;RETURN HERE IF ERROR
017724  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
017726  000137  020776                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
017732                          50$:

                                ;VERIFY NO ERRORS DURING FORMAT
017732  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
017736  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
017740  000240                          NOP                     ;RETURN HERE IF ERROR
017742  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
017744  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
017746  000137  020776                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
017752                          60$:

                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
017752  012737  020012  001122          MOV     #70$,$LPADR
017760  012737  020012  001124          MOV     #70$,$LPERR
017766  013737  001434  021000          MOV     RMDCO,360$
017774  012737  020004  001210          MOV     #65$,$ESCAPE    ;;ESCAPE TO 65$ ON ERROR
020002  000413                          BR      80$                     ;SKIP TO WRITE OPERATION

020004  013737  021000  001434  65$:    MOV     360$,RMDCO              ;RESTORE CYLINDER
                                ;;**************************************************************
                                ;LOOP #2        WRITE,READ

020012                          70$:
```

```
                                        ;PREPARE DEVICE FOR WRITE OPERATION
020012  004737  040020                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
020016  054130                                .WORD   054130  ;TASK DESCRIPTOR
020020  000404                                BR      80$                     ;GO TO 80$ IF NO ERROR
020022  000240                                NOP                     ;RETURN HERE IF ERROR
020024  104000                                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
020026  000137  020776                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
020032                                  80$:
020032  013737  001434  021000                MOV     RMDCO,360$              ;SAVE CYLINER
020040  012737  001466  001434                MOV     #822.,RMDCO             ;SEEK TO LAST CYLINER
020046  012737  000005  001400                MOV     #SEEK!GO,RMCS10         ;WRITE SEEK COMMAND
020054  004737  044006                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
020060  000404                                BR      90$                     ;GO TO 90$ IF NO ERROR
020062  000240                                NOP                     ;RETURN HERE IF ERROR
020064  104000                                ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
020066  000137  020776                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
020072                                  90$:

                                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
020072  004737  043452                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
020076  004737  044346                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
020102  004737  043536                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
020106  000404                                BR      100$                    ;GO TO 100$ IF NO ERROR
020110  000240                                NOP                     ;RETURN HERE IF ERROR
020112  104000                                ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
020114  000137  020776                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
020120                                  100$:
020120  004737  051440                        JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
020124  000405                                BR      110$                    ;GO TO 110$ IF NO ERROR
020126  000240                                NOP                     ;RETURN HERE IF ERROR
020130  104000                                ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
020132  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
020134  000137  020776                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
020140  013737  021000  001434  110$:         MOV     360$,RMDCO              ;RESTORE CYLINDER
020146  012737  000000  001210                MOV     #0,$ESCAPE      ;;ESCAPE TO 0 ON ERROR

020154                                  120$:

                                        ;WRITE DATA TO THE DRIVE
020154  012737  177400  001402                MOV     #<-C256.+1>,RMWCO       ;CHANGE WORD COUNT
020162  012737  107554  001404                MOV     #BUFONE+4,RMBAO         ;CHANGE BUS ADDRESS
020170  012737  000061  001400                MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
020176  012702  001535                        MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
020202  112722  000006                        MOVB    #RMDA,(R2)+
020206  112722  000034                        MOVB    #RMDC,(R2)+
020212  112722  000032                        MOVB    #RMOF,(R2)+
020216  112722  000004                        MOVB    #RMBA,(R2)+
020222  112722  000002                        MOVB    #RMWC,(R2)+
020226  112722  000000                        MOVB    #RMCS1,(R2)+
020232  112722  000200                        MOVB    #200,(R2)+              ;TERMINATE TABLE

020236  004737  044006                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
020242  000404                                BR      130$                    ;GO TO 130$ IF NO ERROR
020244  000240                                NOP                     ;RETURN HERE IF ERROR
020246  104000                                ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
```

```
                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
                                130$:
                                ;WAIT FOR WRITE COMMAND TO COMPLETE
                                        JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
                                ;GO READ STATUS FOR WRITE COMMAND
                                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
                                        BR      140$                    ;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$:
                                ;CHANGE LOOP ADDRESSES
                                        MOV     #190$,$LPADR
                                        MOV     #190$,$LPERR
                                        MOV     RMDCO,360$
                                        MOV     #185$,$ESCAPE   ;;ESCAPE TO 185$ ON ERROR
                                        BR      200$                    ;SKIP TO NEXT OPERATION
                                185$:   MOV     360$,RMDCO              ;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$:
                                ;
                                        MOV     RMDC0,360$                      ;SAVE CYLINDER
                                        MOV     #822.,RMDC0                     ;SEEK TO LAST CYLINDER
                                        MOV     #SEEK!GO,RMCS10                 ;WRITE SEEK COMMAND
                                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
                                        BR      210$                            ;GO TO 210$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY PUT SUBROUTINE
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                210$:
                                ;
                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
                                        JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
                                        JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
                                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
                                        BR      220$                            ;GO TO 220$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY GET SUBROUTINE
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                220$:
                                        JSR     PC,SEKSTS               ;GO VERIFY RESULTS OF SEEK OPERATION
                                        BR      230$                            ;GO TO 230$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY SEKSTS SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
                                230$:   MOV     360$,RMDC0                      ;RESTORE CYLINDER
                                        MOV     #0,$ESCAPE              ;;ESCAPE TO 0 ON ERROR
                                ;
                                240$:
                                ;
                                ;READ DATA FROM DEVICE
                                        MOV     #RD!GO,RMCS10                   ;READ DATA COMMAND
                                        MOV     #BUFTW0+4,RMBA0                 ;LOAD STARTING BUFFER ADDRESS
                                        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$:
                                ;
                                ;
                                ;WAIT FOR READ OPERATION TO COMPLETE
```

```
5716  020652  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5717
5718                                  ;GO READ STATUS FOR READ OPERATION
5719  020656  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5720  020662  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
5721  020664  000240                          NOP                     ;RETURN HERE IF ERROR
5722  020666  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5723  020670  000137  020776                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5724  020674                          260$:
5725
5726                                  ;CHECK FOR ERRORS DURING READ OPERATION
5727  020674  004737  044532                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
5728  020700  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
5729  020702  000240                          NOP                     ;RETURN HERE IF ERROR
5730  020704  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5731  020706  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5732  020710  000137  020776                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5733  020714                          270$:
5734  020714  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
5735  020720  000405                          BR      280$                    ;GO TO 280$ IF NO ERROR
5736  020722  000240                          NOP                     ;RETURN HERE IF ERROR
5737  020724  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5738  020726  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5739  020730  000137  020776                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5740  020734                          280$:
5741
5742  020734                          290$:
5743  020734  004737  045364                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
5744  020740  000405                          BR      300$                    ;GO TO 300$ IF NO ERROR
5745  020742  000240                          NOP                     ;RETURN HERE IF ERROR
5746  020744  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5747  020746  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5748  020750  000137  020776                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5749  020754                          300$:
5750  020754  004737  043104                  JSR     PC,CMPBUF               ;GO COMPARE WRITE, READ DATA BUFFERS
5751  020760  107554                          .WORD   BUFONE+4                        ;STARTING ADDRESS OF WRITE BUFFER
5752  020762  110560                          .WORD   BUFTWO+4                        ;STARTING ADDDRESS OF READ BUFFER
5753  020764  000404                          BR      310$                    ;GO TO 310$ IF NO ERROR
5754  020766  000240                          NOP                     ;RETURN HERE IF ERROR
5755  020770  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
5756  020772  000137  020776                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5757  020776                          310$:
5758
5759  020776                          350$:
5760  020776  000401                          BR      370$
5761  021000  000000                  360$:   .WORD
5762  021002                          370$:
5763                                  ;;*****************************************************************
5764                                  ;*TEST 14        WRITE, WRITE CHECK W/ HEAD SWITCHING
5765                                  ;;*****************************************************************
5766  021002                          TST14:
5767  021002  000004                          SCOPE                   ;SCOPE CALL
5768  021004  000240                          NOP                     ;START OF TEST
5769  021006  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5770  021012  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5771  021016  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
```

```
5772  021022  012737  000014  001226          MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5773
5774                                          ;;*****************************************************************
5775                                          ;LOOP #1        FORMAT,WRITE,READ
5776
5777  021030                                  10$:
5778
5779                                          ;PREPARE THE DEVICE FOR FORMAT OPERATION
5780  021030  004737  040020                         JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5781  021034  054130                                 .WORD   054130  ;TASK DESCRIPTOR
5782  021036  000404                                 BR      20$                     ;GO TO 20$ IF NO ERROR
5783  021040  000240                                 NOP                     ;RETURN HERE IF ERROR
5784  021042  104000                                 ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5785  021044  000137  022022                         JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5786  021050                                  20$:
5787
5788                                          ;LOAD PARAMETERS AND GENERATE DATA BUFFER
5789  021050  012737  000000  001434                 MOV     #0,RMDC0                ;CYLINDER = 0
5790  021056  012737  000037  001406                 MOV     #037,RMDA0              ;TRACK = 0, SECTOR = 31
5791  021064  012737  107550  001404                 MOV     #BUFONE,RMBA0           ;BUS ADDRESS
5792  021072  012737  176774  001402                 MOV     #<^C<2*<256.+2>>+1>,RMWC0               ;WORD COUNT
5793  021100  012737  010000  001432                 MOV     #FMT16,RM0F0            ;16 BIT FORMAT
5794  021106  012737  000063  001400                 MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
5795
5796                                          ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
5797
5798  021114  004737  040734                         JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
5799  021120  000405                                 BR      25$                     ;GO TO 25$ IF NO ERROR
5800  021122  104401  067466                         TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
5801  021126  104000                                 ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
5802  021130  000137  022022                         JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5803  021134                                  25$:
5804  021134  012737  071046  001174                 MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
5805  021142  012737  000001  001176                 MOV     #1,$TMP1                ;RANGE OF PATTERN
5806  021150  004737  042640                         JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
5807
5808                                          ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
5809  021154  012702  001535                         MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
5810  021160  112722  000034                         MOVB    #RMDC,(R2)+
5811  021164  112722  000006                         MOVB    #RMDA,(R2)+
5812  021170  112722  000004                         MOVB    #RMBA,(R2)+
5813  021174  112722  000002                         MOVB    #RMWC,(R2)+
5814  021200  112722  000032                         MOVB    #RMOF,(R2)+
5815  021204  112722  000000                         MOVB    #RMCS1,(R2)+
5816  021210  112712  000200                         MOVB    #200,(R2)               ;TERMINATE TABLE
5817  021214                                  30$:
5818
5819                                          ;FORMAT THE DRIVE
5820  021214  004737  044006                         JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5821  021220  000404                                 BR      40$                     ;GO TO 40$ IF NO ERROR
5822  021222  000240                                 NOP                     ;RETURN HERE IF ERROR
5823  021224  104000                                 ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5824  021226  000137  022022                         JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5825  021232                                  40$:
5826
5827                                          ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
```

```
5828  021232  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5829
5830                                  ;WAIT FOR THE FORMAT TO COMPLETE
5831  021236  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5832
5833                                  ;GO READ STATUS FOR FORMAT OPERATION
5834  021242  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5835  021246  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
5836  021250  000240                          NOP                     ;RETURN HERE IF ERROR
5837  021252  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5838  021254  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5839  021260                          50$:
5840
5841                                  ;VERIFY NO ERRORS DURING FORMAT
5842  021260  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5843  021264  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
5844  021266  000240                          NOP                     ;RETURN HERE IF ERROR
5845  021270  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5846  021272  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5847  021274  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5848  021300                          60$:
5849
5850                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
5851  021300  012737  021360  001122          MOV     #70$,$LPADR
5852  021306  012737  021360  001124          MOV     #70$,$LPERR
5853                                  ;REGENERATE BUFFER
5854  021314  012737  177000  001402          MOV     #<^C<2*256.>+1>,RMWCO   ;CHANGE WORD COUNT
5855  021322  012737  107550  001404          MOV     #BUFONE,RMBAO           ;CHANGE BUS ADDRESS
5856  021330  012737  071110  001174          MOV     #ZEROS,$TMPO            ;STARTING ADDRESS
5857  021336  012737  000001  001176          MOV     #1,$TMP1                ;RANGE
5858  021344  012737  000060  001400          MOV     #WD,RMCS10              ;WRITE DATA
5859  021352  004737  042640                  JSR     PC,GENBUF               ;GENERATE BUFFER
5860  021356  000410                          BR      80$                     ;SKIP TO WRITE OPERATION
5861
5862                                  ;;*****************************************************************
5863                                  ;LOOP #2        WRITE,READ
5864
5865  021360                          70$:
5866
5867
5868                                  ;PREPARE DEVICE FOR WRITE OPERATION
5869  021360  004737  040020                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
5870  021364  054130                          .WORD   054130  ;TASK DESCRIPTOR
5871  021366  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
5872  021370  000240                          NOP                     ;RETURN HERE IF ERROR
5873  021372  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5874  021374  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5875  021400                          80$:
5876
5877
5878  021400                          120$:
5879
5880                                  ;WRITE DATA TO THE DRIVE
5881  021400  012737  000061  001400          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
5882  021406  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5883  021412  112722  000006                  MOVB    #RMDA,(R2)+
```

```
5884  021416  112722  000034                  MOVB    #RMDC,(R2)+
5885  021422  112722  000032                  MOVB    #RMOF,(R2)+
5886  021426  112722  000004                  MOVB    #RMBA,(R2)+
5887  021432  112722  000002                  MOVB    #RMWC,(R2)+
5888  021436  112722  000000                  MOVB    #RMCS1,(R2)+
5889  021442  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
5890
5891  021446  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5892  021452  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
5893  021454  000240                          NOP                     ;RETURN HERE IF ERROR
5894  021456  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5895  021460  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5896  021464                          130$:
5897
5898                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5899  021464  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
5900
5901                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
5902  021470  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
5903
5904                                  ;GO READ STATUS FOR WRITE COMMAND
5905  021474  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5906  021500  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
5907  021502  000240                          NOP                     ;RETURN HERE IF ERROR
5908  021504  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5909  021506  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5910  021512                          140$:
5911
5912                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
5913  021512  004737  044532                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
5914  021516  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
5915  021520  000240                          NOP                     ;RETURN HERE IF ERROR
5916  021522  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5917  021524  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5918  021526  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5919  021532                          150$:
5920  021532  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
5921  021536  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
5922  021540  000240                          NOP                     ;RETURN HERE IF ERROR
5923  021542  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5924  021544  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5925  021546  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5926  021552                          160$:
5927
5928  021552                          170$:
5929  021552  004737  045364                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
5930  021556  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
5931  021560  000240                          NOP                     ;RETURN HERE IF ERROR
5932  021562  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5933  021564  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5934  021566  000137  022022                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5935  021572                          180$:
5936
5937                                  ;CHANGE LOOP ADDRESSES
5938  021572  012737  021610  001122          MOV     #190$,$LPADR
5939  021600  012737  021610  001124          MOV     #190$,$LPERR
```

```
5940  021606  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
5941
5942                                  ;;*****************************************************************
5943                                  ;LOOP #3        READ
5944
5945  021610                          190$:
5946
5947                                  ;PREPARE DEVICE FOR READ OPERATION
5948  021610  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5949  021614  054130                        .WORD   054130  ;TASK DESCRIPTOR
5950  021616  000404                        BR      200$                    ;GO TO 200$ IF NO ERROR
5951  021620  000240                        NOP                     ;RETURN HERE IF ERROR
5952  021622  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5953  021624  000137  022022                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5954  021630                          200$:
5955
5956  021630                          240$:
5957
5958                                  ;READ DATA FROM DEVICE
5959  021630  012737  000051  001400        MOV     #WCD!GO,RMCS10          ;READ DATA COMMAND
5960  021636  012702  001535                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
5961  021642  112722  000006                MOVB    #RMDA,(R2)+
5962  021646  112722  000032                MOVB    #RMOF,(R2)+
5963  021652  112722  000034                MOVB    #RMDC,(R2)+
5964  021656  112722  000004                MOVB    #RMBA,(R2)+
5965  021662  112722  000002                MOVB    #RMWC,(R2)+
5966  021666  112722  000000                MOVB    #RMCS1,(R2)+
5967  021672  112712  000200                MOVB    #200,(R2)
5968
5969  021676  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
5970  021702  000404                        BR      250$                    ;GO TO 250$ IF NO ERROR
5971  021704  000240                        NOP                     ;RETURN HERE IF ERROR
5972  021706  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
5973  021710  000137  022022                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5974  021714                          250$:
5975
5976                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
5977  021714  004737  043452                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
5978
5979                                  ;WAIT FOR READ OPERATION TO COMPLETE
5980  021720  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
5981
5982                                  ;GO READ STATUS FOR READ OPERATION
5983  021724  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
5984  021730  000404                        BR      260$                    ;GO TO 260$ IF NO ERROR
5985  021732  000240                        NOP                     ;RETURN HERE IF ERROR
5986  021734  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
5987  021736  000137  022022                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
5988  021742                          260$:
5989
5990                                  ;CHECK FOR ERRORS DURING READ OPERATION
5991  021742  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5992  021746  000405                        BR      270$                    ;GO TO 270$ IF NO ERROR
5993  021750  000240                        NOP                     ;RETURN HERE IF ERROR
5994  021752  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5995  021754  004736                        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 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     #BUF0NE,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
                                25$:
                                        MOV     #ZEROS,$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
                                50$:

                                ;VERIFY NO ERRORS DURING FORMAT
                                        JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
                                        BR      60$                     ;GO TO 60$ 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
                                60$:

                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
                                        MOV     #70$,$LPADR
                                        MOV     #70$,$LPERR
                                ;REGENERATE BUFFER
                                        MOV     #WD,RMCS10              ;WRITE DATA
```

```
6108  022342  012737  177000  001402          MOV     #<^C<2*256.>+1>,RMWCO    ;CHANGE WORD COUNT
6109  022350  012737  107550  001404          MOV     #BUFONE,RMBAO            ;CHANGE BUS ADDRSS
6110  022356  012737  071046  001174          MOV     #ONES,$TMPO              ;PATTERN ADDRESS
6111  022364  012737  000001  001176          MOV     #1,$TMP1                 ;PATTERN RANGE
6112  022372  004737  042640                  JSR     PC,GENBUF
6113  022376  000410                          BR      80$                      ;SKIP TO WRITE OPERATION
6114
6115                          ;;*****************************************************************
6116                          ;LOOP #2        WRITE,READ
6117
6118  022400                  70$:
6119
6120
6121                          ;PREPARE DEVICE FOR WRITE OPERATION
6122  022400  004737  040020          JSR     PC,TSTPRP        ;PREPARE DEVICE FOR TEST
6123  022404  054130                  .WORD   054130  ;TASK DESCRIPTOR
6124  022406  000404                  BR      80$                      ;GO TO 80$ IF NO ERROR
6125  022410  000240                  NOP                      ;RETURN HERE IF ERROR
6126  022412  104000                  ERROR                    ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6127  022414  000137  023042          JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
6128  022420                  80$:
6129
6130
6131  022420                  120$:
6132
6133                          ;WRITE DATA TO THE DRIVE
6134  022420  012737  000061  001400          MOV     #WD!GO,RMCS10            ;WRITE DATA COMMAND
6135  022426  012702  001535                  MOV     #PUTINX,R2               ;LOAD PUT REGISTER INDEX TABLE
6136  022432  112722  000006                  MOVB    #RMDA,(R2)+
6137  022436  112722  000034                  MOVB    #RMDC,(R2)+
6138  022442  112722  000032                  MOVB    #RMOF,(R2)+
6139  022446  112722  000004                  MOVB    #RMBA,(R2)+
6140  022452  112722  000002                  MOVB    #RMWC,(R2)+
6141  022456  112722  000000                  MOVB    #RMCS1,(R2)+
6142  022462  112722  000200                  MOVB    #200,(R2)+               ;TERMINATE TABLE
6143
6144  022466  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6145  022472  000404                  BR      130$                     ;GO TO 130$ IF NO ERROR
6146  022474  000240                  NOP                      ;RETURN HERE IF ERROR
6147  022476  104000                  ERROR                    ;ERROR # DEFINED BY PUT SUBROUTINE
6148  022500  000137  023042          JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
6149  022504                  130$:
6150
6151                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
6152  022504  004737  043452          JSR     PC,GETSTS        ;GO TO GETSTS SUBROUTINE
6153
6154                          ;WAIT FOR WRITE COMMAND TO COMPLETE
6155  022510  004737  044346          JSR     PC,TIMOUT        ;GO TO TIMOUT SUBROUTINE
6156
6157                          ;GO READ STATUS FOR WRITE COMMAND
6158  022514  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6159  022520  000404                  BR      140$                     ;GO TO 140$ IF NO ERROR
6160  022522  000240                  NOP                      ;RETURN HERE IF ERROR
6161  022524  104000                  ERROR                    ;ERROR # DEFINED BY GET SUBROUTINE
6162  022526  000137  023042          JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
6163  022532                  140$:
```

```
                                ;CHECK FOR ERRORS DURING WRITE OPERATION
022532  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
022536  000405                      BR      150$                    ;GO TO 150$ IF NO ERROR
022540  000240                      NOP                     ;RETURN HERE IF ERROR
022542  104000                      ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
022544  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
022546  000137  023042              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
022552                          150$:
022552  004737  057036              JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
022556  000405                      BR      160$                    ;GO TO 160$ IF NO ERROR
022560  000240                      NOP                     ;RETURN HERE IF ERROR
022562  104000                      ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
022564  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
022566  000137  023042              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
022572                          160$:
                                
022572                          170$:
022572  004737  045364              JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
022576  000405                      BR      180$                    ;GO TO 180$ IF NO ERROR
022600  000240                      NOP                     ;RETURN HERE IF ERROR
022602  104000                      ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
022604  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
022606  000137  023042              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
022612                          180$:
                                
                                ;CHANGE LOOP ADDRESSES
022612  012737  022630  001122        MOV     #190$,$LPADR
022620  012737  022630  001124        MOV     #190$,$LPERR
022626  000410                      BR      200$                    ;SKIP TO NEXT OPERATION
                                
                                ;;*****************************************************************
                                ;LOOP #3        READ
                                
022630                          190$:
                                
                                ;PREPARE DEVICE FOR READ OPERATION
022630  004737  040020              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
022634  054130                      .WORD   054130  ;TASK DESCRIPTOR
022636  000404                      BR      200$                    ;GO TO 200$ IF NO ERROR
022640  000240                      NOP                     ;RETURN HERE IF ERROR
022642  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
022644  000137  023042              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
022650                          200$:
                                
022650                          240$:
                                
                                ;READ DATA FROM DEVICE
022650  012737  000051  001400        MOV     #WCD!GO,RMCS10          ;READ DATA COMMAND
022656  012702  001535              MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
022662  112722  000006              MOVB    #RMDA,(R2)+
022666  112722  000032              MOVB    #RMOF,(R2)+
022672  112722  000034              MOVB    #RMDC,(R2)+
022676  112722  000004              MOVB    #RMBA,(R2)+
022702  112722  000002              MOVB    #RMWC,(R2)+
022706  112722  000000              MOVB    #RMCS1,(R2)+
```

```
6220  022712  112712  000200                  MOVB    #200,(R2)
6221
6222  022716  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6223  022722  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
6224  022724  000240                          NOP                     ;RETURN HERE IF ERROR
6225  022726  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6226  022730  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6227  022734                          250$:
6228
6229                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
6230  022734  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6231
6232                                  ;WAIT FOR READ OPERATION TO COMPLETE
6233  022740  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
6234
6235                                  ;GO READ STATUS FOR READ OPERATION
6236  022744  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6237  022750  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
6238  022752  000240                          NOP                     ;RETURN HERE IF ERROR
6239  022754  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6240  022756  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6241  022762                          260$:
6242
6243                                  ;CHECK FOR ERRORS DURING READ OPERATION
6244  022762  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6245  022766  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
6246  022770  000240                          NOP                     ;RETURN HERE IF ERROR
6247  022772  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6248  022774  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6249  022776  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6250  023002                          270$:
6251  023002  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6252  023006  000405                          BR      280$                    ;GO TO 280$ IF NO ERROR
6253  023010  000240                          NOP                     ;RETURN HERE IF ERROR
6254  023012  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6255  023014  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6256  023016  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6257  023022                          280$:
6258
6259  023022                          290$:
6260  023022  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6261  023026  000405                          BR      300$                    ;GO TO 300$ IF NO ERROR
6262  023030  000240                          NOP                     ;RETURN HERE IF ERROR
6263  023032  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6264  023034  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6265  023036  000137  023042                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6266  023042                          300$:
6267
6268  023042                          350$:
6269                                  ;;*****************************************************************
6270                                  ;*TEST 16        WRITE, READ W/ HCE ERROR
6271                                  ;;*****************************************************************
6272  023042                          TST16:
6273  023042  000004                          SCOPE                   ;SCOPE CALL
6274  023044  000240                          NOP                     ;START OF TEST
6275  023046  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
```

```
6276  023052  013700  001276            MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6277  023056  013701  001450            MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6278  023062  012737  000016  001226    MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6279
6280                              ;;*****************************************************************
6281                              ;LOOP #1        FORMAT,WRITE,READ
6282
6283  023070  012704  000000            MOV     #0,R4                           ;R4=HEADER WORD
6284                              ;;*****************************************************************
6285                              ;NOTE:  BSE NOT TESTED
6286  023074  012703  020000      10$:  MOV     #BIT13,R3                       ;R3=HCE BIT
6287  023100                      15$:
6288
6289                              ;PREPARE THE DEVICE FOR FORMAT OPERATION
6290  023100  004737  040020            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6291  023104  054130                    .WORD   054130  ;TASK DESCRIPTOR
6292  023106  000404                    BR      20$                             ;GO TO 20$ IF NO ERROR
6293  023110  000240                    NOP                             ;RETURN HERE IF ERROR
6294  023112  104000                    ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6295  023114  000137  024724            JMP     370$                            ;GO TO 370$ IF ERROR WAS FOUND
6296  023120                      20$:
6297
6298                              ;LOAD PARAMETERS AND GENERATE DATA BUFFER
6299  023120  012737  000000  001434    MOV     #0,RMDC0                        ;CYLINDER = 0
6300  023126  012737  000000  001406    MOV     #0,RMDA0                        ;TRACK = 0, SECTOR = 0
6301  023134  012737  107550  001404    MOV     #BUFONE,RMBA0                   ;BUS ADDRESS
6302  023142  012737  177376  001402    MOV     #<^C<2+256.>+1>,RMWC0           ;WORD COUNT
6303  023150  012737  010000  001432    MOV     #FMT16,RMOF0                    ;16 BIT FORMAT
6304  023156  012737  000063  001400    MOV     #WH!GO,RMCS10                   ;WRITE HEADER AND DATA COMMAND
6305
6306                              ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
6307
6308  023164  004737  040734            JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
6309  023170  000405                    BR      25$                             ;GO TO 25$ IF NO ERROR
6310  023172  104401  067466            TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
6311  023176  104000                    ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
6312  023200  000137  024536            JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
6313  023204                      25$:
6314  023204  012737  071110  001174    MOV     #ZEROS,$TMP0                    ;STARTING ADDRESS OF PATTERN
6315  023212  012737  000001  001176    MOV     #1,$TMP1                        ;RANGE OF PATTERN
6316  023220  004737  042640            JSR     PC,GENBUF                       ;GO GENERATE BUFFER FOR FORMAT
6317
6318                              ;CHANGE HEADER WORD TO FORCE HCE DURING WRITE & READ
6319  023224  030364  107550            BIT     R3,BUFONE(R4)                   ;SET OR RESET FOR HCE??
6320  023230  001403                    BEQ     26$                             ;
6321  023232  040364  107550            BIC     R3,BUFONE(R4)                   ;RESET BIT FOR HCE
6322  023236  000402                    BR      27$
6323  023240  050364  107550      26$:  BIS     R3,BUFONE(R4)                   ;SET FOR HCE
6324  023244                      27$:
6325                              ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
6326  023244  012702  001535            MOV     #PUTINX,R2                      ;R2 = BYTE ENTRY POSITION
6327  023250  112722  000034            MOVB    #RMDC,(R2)+
6328  023254  112722  000006            MOVB    #RMDA,(R2)+
6329  023260  112722  000004            MOVB    #RMBA,(R2)+
6330  023264  112722  000002            MOVB    #RMWC,(R2)+
6331  023270  112722  000032            MOVB    #RMOF,(R2)+
```

```
6332  023274  112722  000000          MOVB    #RMCS1,(R2)+
6333  023300  112712  000200          MOVB    #200,(R2)               ;TERMINATE TABLE
6334  023304                      30$:
6335
6336                              ;FORMAT THE DRIVE
6337  023304  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6338  023310  000404                  BR      40$                     ;GO TO 40$ IF NO ERROR
6339  023312  000240                  NOP                     ;RETURN HERE IF ERROR
6340  023314  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6341  023316  000137  024536          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6342  023322                      40$:
6343
6344                              ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
6345  023322  004737  043452          JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
6346
6347                              ;WAIT FOR THE FORMAT TO COMPLETE
6348  023326  004737  044346          JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
6349
6350                              ;GO READ STATUS FOR FORMAT OPERATION
6351  023332  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6352  023336  000404                  BR      50$                     ;GO TO 50$ IF NO ERROR
6353  023340  000240                  NOP                     ;RETURN HERE IF ERROR
6354  023342  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6355  023344  000137  024536          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6356  023350                      50$:
6357
6358                              ;VERIFY NO ERRORS DURING FORMAT
6359  023350  004737  057036          JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6360  023354  000405                  BR      60$                     ;GO TO 60$ IF NO ERROR
6361  023356  000240                  NOP                     ;RETURN HERE IF ERROR
6362  023360  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6363  023362  004736                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6364  023364  000137  024536          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6365  023370                      60$:
6366
6367                              ;MOVE LOOP ADDRESSES TO NEXT OPERATION
6368  023370  012737  023406  001122      MOV     #70$,$LPADR
6369  023376  012737  023406  001124      MOV     #70$,$LPERR
6370  023404  000410                  BR      80$                     ;SKIP TO WRITE OPERATION
6371
6372                              ;;*****************************************************************
6373                              ;LOOP #2        WRITE,READ
6374
6375  023406                      70$:
6376
6377
6378                              ;PREPARE DEVICE FOR WRITE OPERATION
6379  023406  004737  040020          JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
6380  023412  054130                  .WORD   054130  ;TASK DESCRIPTOR
6381  023414  000404                  BR      80$                     ;GO TO 80$ IF NO ERROR
6382  023416  000240                  NOP                     ;RETURN HERE IF ERROR
6383  023420  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6384  023422  000137  024536          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6385  023426                      80$:
6386
6387
```

```
6388  023426                          120$:
6389
6390                                  ;WRITE DATA TO THE DRIVE
6391  023426  012737  107554  001404          MOV     #BUFONE+4,RMBA0          ;CHANGE BUS ADDRESS
6392  023434  012737  177400  001402          MOV     #<^C256.+1>,RMWC0        ;CHANGE WORD COUNT
6393  023442  012737  000061  001400          MOV     #WD!GO,RMCS10            ;WRITE DATA COMMAND
6394  023450  012702  001535                  MOV     #PUTINX,R2               ;LOAD PUT REGISTER INDEX TABLE
6395  023454  112722  000006                  MOVB    #RMDA,(R2)+
6396  023460  112722  000034                  MOVB    #RMDC,(R2)+
6397  023464  112722  000032                  MOVB    #RMOF,(R2)+
6398  023470  112722  000004                  MOVB    #RMBA,(R2)+
6399  023474  112722  000002                  MOVB    #RMWC,(R2)+
6400  023500  112722  000000                  MOVB    #RMCS1,(R2)+
6401  023504  112722  000200                  MOVB    #200,(R2)+               ;TERMINATE TABLE
6402
6403  023510  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6404  023514  000404                          BR      130$                     ;GO TO 130$ IF NO ERROR
6405  023516  000240                          NOP                      ;RETURN HERE IF ERROR
6406  023520  104000                          ERROR                    ;ERROR # DEFINED BY PUT SUBROUTINE
6407  023522  000137  024536                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
6408  023526                          130$:
6409
6410                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
6411  023526  004737  043452                  JSR     PC,GETSTS        ;GO TO GETSTS SUBROUTINE
6412
6413                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
6414  023532  004737  044346                  JSR     PC,TIMOUT        ;GO TO TIMOUT SUBROUTINE
6415
6416                                  ;GO READ STATUS FOR WRITE COMMAND
6417  023536  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6418  023542  000404                          BR      140$                     ;GO TO 140$ IF NO ERROR
6419  023544  000240                          NOP                      ;RETURN HERE IF ERROR
6420  023546  104000                          ERROR                    ;ERROR # DEFINED BY GET SUBROUTINE
6421  023550  000137  024536                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
6422  023554                          140$:
6423
6424                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
6425  023554  004737  044532                  JSR     PC,PRIERR        ;GO CHECK FOR PRIMARY ERRORS
6426  023560  000405                          BR      145$                     ;GO TO 145$ IF NO ERROR
6427  023562  000240                          NOP                      ;RETURN HERE IF ERROR
6428  023564  104000                          ERROR                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
6429  023566  004736                          JSR     PC,@(SP)+        ;GO BACK FOR MORE ERROR CHECKS
6430  023570  000137  024536                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
6431  023574                          145$:
6432
6433                                  ;DECODE THE TYPE OF ERROR PRODUCED BY THE BIT POSITION OF R3
6434  023574  005704                          TST     R4                       ;IS THIS THE FIRST HEADER WORD ?
6435  023576  001006                          BNE     146$                     ;NO !!
6436  023600  032703  010000                  BIT     #FMT16,R3                ;IS THIS A FORMAT ERROR ??
6437  023604  001034                          BNE     155$                     ;YES !!
6438  023606  032703  140000                  BIT     #MSE!USE,R3              ;IS THIS A BAD SECTOR ERROR ??
6439  023612  001056                          BNE     165$                     ;YES !!
6440  023614                          146$:
6441
6442                                  ;VERIFY THAT A HEADER COMPARE ERROR WAS DETECTED
6443  023614  032737  000200  001344          BIT     #HCE,RMER1I              ;WAS HCE DETECTED ??
```

```
6444  023622  001077                          BNE     175$                    ;YES !!
6445                                  ;HCE WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6446  023624  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6447  023630  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
6448  023632  000240                          NOP                             ;RETURN HERE IF ERROR
6449  023634  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
6450  023636  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6451  023640  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6452  023644                          150$:
6453  023644  013737  001344  001142          MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
6454  023652  012737  000200  001140          MOV     #HCE,$GDDAT             ;EXPECTED STATUS
6455  023660  010437  001174                  MOV     R4,$TMP0                ;R4 = HEADER WORD NUMBER
6456  023664  010337  001176                  MOV     R3,$TMP1                ;R3 = BIT POSIITON
6457  023670  104344                          ERROR   344                     ;HCE NOT DETECTED
6458  023672  000137  024536                  JMP     350$
6459  023676                          155$:
6460
6461                                  ;VERIFY THAT A FORMAT ERROR WAS DETECTED
6462  023676  032737  000020  001344          BIT     #FER,RMER1I             ;WAS FER DETECTED ??
6463  023704  001046                          BNE     175$                    ;YES !!
6464                                  ;FER WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6465  023706  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6466  023712  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
6467  023714  000240                          NOP                             ;RETURN HERE IF ERROR
6468  023716  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
6469  023720  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6470  023722  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6471  023726                          160$:
6472  023726  012737  000020  001140          MOV     #FER,$GDDAT             ;EXPECTED STATUS
6473  023734  013737  001344  001142          MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
6474  023742  104343                          ERROR   343                     ;FER NOT DETECTED
6475  023744  000137  024536                  JMP     350$
6476  023750                          165$:
6477
6478                                  ;VERIFY THAT A BAD SECTOR ERROR WAS DETECTED
6479  023750  032737  100000  001372          BIT     #BSE,RMER2I             ;WAS BSE DETECTED ??
6480  023756  001021                          BNE     175$                    ;YES !!
6481                                  ;BSE WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
6482  023760  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6483  023764  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
6484  023766  000240                          NOP                             ;RETURN HERE IF ERROR
6485  023770  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
6486  023772  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6487  023774  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6488  024000                          170$:
6489  024000  013737  001372  001142          MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
6490  024006  012737  100000  001140          MOV     #BSE,$GDDAT             ;EXPECTED STAIUS
6491  024014  104345                          ERROR   345                     ;BSE NOTDETECTED
6492  024016  000137  024536                  JMP     350$
6493  024022                          175$:
6494  024022  004737  045364                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
6495  024026  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
6496  024030  000240                          NOP                             ;RETURN HERE IF ERROR
6497  024032  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
6498  024034  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6499  024036  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
```

```
6500  024042                          180$:
6501
6502                                  ;CHANGE LOOP ADDRESSES
6503  024042  012737  024056  001122          MOV     #190$,$LPADR
6504  024050  012737  024056  001124          MOV     #190$,$LPERR
6505
6506                                  ;;*****************************************************************
6507                                  ;LOOP #3        READ
6508
6509  024056                          190$:
6510
6511                                  ;PREPARE DEVICE FOR READ OPERATION
6512  024056  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6513  024062  054130                          .WORD   054130  ;TASK DESCRIPTOR
6514  024064  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
6515  024066  000240                          NOP                     ;RETURN HERE IF ERROR
6516  024070  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6517  024072  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6518  024076                          200$:
6519
6520  024076                          240$:
6521
6522                                  ;READ DATA FROM DEVICE
6523  024076  012737  110560  001404          MOV     #BUFTWO+4,RMBA0         ;CHANGE BUS ADDRESS
6524  024104  012737  177400  001402          MOV     #<^C256.+1>,RMWC0       ;CHANGE WORD COUNT
6525  024112  012737  000071  001400          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
6526  024120  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
6527  024124  112722  000006                  MOVB    #RMDA,(R2)+
6528  024130  112722  000032                  MOVB    #RMOF,(R2)+
6529  024134  112722  000034                  MOVB    #RMDC,(R2)+
6530  024140  112722  000004                  MOVB    #RMBA,(R2)+
6531  024144  112722  000002                  MOVB    #RMWC,(R2)+
6532  024150  112722  000000                  MOVB    #RMCS1,(R2)+
6533  024154  112712  000200                  MOVB    #200,(R2)
6534
6535  024160  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6536  024164  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
6537  024166  000240                          NOP                     ;RETURN HERE IF ERROR
6538  024170  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6539  024172  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6540  024176                          250$:
6541
6542                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
6543  024176  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6544
6545                                  ;WAIT FOR READ OPERATION TO COMPLETE
6546  024202  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
6547
6548                                  ;GO READ STATUS FOR READ OPERATION
6549  024206  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6550  024212  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
6551  024214  000240                          NOP                     ;RETURN HERE IF ERROR
6552  024216  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6553  024220  000137  024536                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6554  024224                          260$:
6555
```

```
                                ;CHECK FOR ERRORS DURING READ OPERATION
024224  004737  044532          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
024230  000405                  BR      265$            ;GO TO 265$ IF NO ERROR
024232  000240                  NOP                     ;RETURN HERE IF ERROR
024234  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
024236  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
024240  000137  024536          JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
024244                  265$:
                                ;DECODE THE TYPE OF ERROR PRODUCED BY THE BIT POSITION OF R3
024244  005704                  TST     R4              ;IS THIS THE FIRST HEADER WORD ?
024246  001006                  BNE     266$            ;NO !!
024250  032703  010000          BIT     #FMT16,R3       ;IS THIS A FORMAT ERROR ??
024254  001034                  BNE     275$            ;YES !!
024256  032703  140000          BIT     #MSE!USE,R3     ;IS THIS A BAD SECTOR ERROR ??
024262  001056                  BNE     285$            ;YES !!
024264                  266$:

                                ;VERIFY THAT A HEADER COMPARE ERROR WAS DETECTED
024264  032737  000200  001344  BIT     #HCE,RMER1I     ;WAS HCE DETECTED ??
024272  001077                  BNE     295$            ;YES !!
                                ;HCE WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
024274  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
024300  000405                  BR      270$            ;GO TO 270$ IF NO ERROR
024302  000240                  NOP                     ;RETURN HERE IF ERROR
024304  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
024306  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
024310  000137  024536          JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
024314                  270$:
024314  013737  001344  001142  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
024322  012737  000200  001140  MOV     #HCE,$GDDAT     ;EXPECTED STATUS
024330  010437  001174          MOV     R4,$TMP0        ;R4 = HEADER WORD NUMBER
024334  010337  001176          MOV     R3,$TMP1        ;R3 = BIT POSITION
024340  104344                  ERROR   344             ;HCE NOT DETECTED
024342  000137  024536          JMP     350$
024346                  275$:

                                ;VERIFY THAT A FORMAT ERROR WAS DETECTED
024346  032737  000020  001344  BIT     #FER,RMER1I     ;WAS FER DETECTED ??
024354  001046                  BNE     295$            ;YES !!
                                ;FER WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
024356  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
024362  000405                  BR      280$            ;GO TO 280$ IF NO ERROR
024364  000240                  NOP                     ;RETURN HERE IF ERROR
024366  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
024370  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
024372  000137  024536          JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
024376                  280$:
024376  012737  000020  001140  MOV     #FER,$GDDAT     ;EXPECTED STATUS
024404  013737  001344  001142  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
024412  104343                  ERROR   343             ;FER NOT DETECTED
024414  000137  024536          JMP     350$
024420                  285$:

                                ;VERIFY THAT A BAD SECTOR ERROR WAS DETECTED
024420  032737  100000  001372  BIT     #BSE,RMER2I     ;WAS BSE DETECTED ??
024426  001021                  BNE     295$            ;YES !!
```

```
                                ;BSE WAS NOT DETECTED - SEE IF THERE WAS ANOTHER ERROR
024430  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
024434  000405                  BR      290$                    ;GO TO 290$ IF NO ERROR
024436  000240                  NOP                     ;RETURN HERE IF ERROR
024440  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
024442  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
024444  000137  024536          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
024450                  290$:
024450  013737  001372  001142  MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
024456  012737  100000  001140  MOV     #BSE,$GDDAT             ;EXPECTED STAIUS
024464  104345                  ERROR   345                     ;BSE NOTDETECTED
024466  000137  000350          JMP     350
024472                  295$:
024472  004737  045364          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
024476  000405                  BR      300$                    ;GO TO 300$ IF NO ERROR
024500  000240                  NOP                     ;RETURN HERE IF ERROR
024502  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
024504  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
024506  000137  024536          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
024512                  300$:

024512  006203                  ASR     R3                      ;SHIFT TO NEXT BIT
024514  001006                  BNE     310$                    ;REPEAT IF NOT DONE
024516  005704                  TST     R4                      ;SECOND HEADER WORD DONE??
024520  001007                  BNE     355$                    ;YES!!
024522  012704  000002          MOV     #2,R4                   ;SETUP FOR SECOND HEADER WORD
                                ;;*****************************************************************
                                ;NOTE:  BSE NOT TESTED
024526  012703  020000          MOV     #BIT13,R3
024532                  310$:
024532  000137  023100          JMP     15$
024536                  350$:
024536  000472                  BR      370$            ;EXIT IF ERROR
024540                  355$:
024540  013737  001476  001434  MOV     ASNDC,RMDC0             ;
024546  013737  001500  001406  MOV     ASNDA,RMDA0             ;
024554  012737  107550  001404  MOV     #BUFONE,RMBA0           ;BUFFER ADDRESS,REFORMAT THE SECTOR
024562  012737  177776  001402  MOV     #-2,RMWC0               ;ONLY TWO HEAD WORDS
024570  012737  010000  001432  MOV     #FMT16,RMOF0            ;ALWAYS IN 16 BITS MODE
024576  012737  000062  001400  MOV     #WH,RMCS10              ;WRITE HEAD AND DATA COMMAND
024604  004737  042640          JSR     PC,GENBUF               ;SET UP THE BUFFER

024610  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
024614  054130                  .WORD   054130  ;TASK DESCRIPTOR
024616  000404                  BR      360$                    ;GO TO 360$ IF NO ERROR
024620  000240                  NOP                     ;RETURN HERE IF ERROR
024622  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
024624  000137  024630          JMP     360$                    ;GO TO 360$ IF ERROR WAS FOUND
024630                  360$:
024630  012737  000063  001400  MOV     #WH!GO,RMCS10           ;FORMAT THE SECTOR
024636  012702  001535          MOV     #PUTINX,R2              ;SET UP THE REGS
024642  112722  000006          MOVB    #RMDA,(R2)+             ;
024646  112722  000032          MOVB    #RMOF,(R2)+             ;
024652  112722  000034          MOVB    #RMDC,(R2)+             ;
024656  112722  000004          MOVB    #RMBA,(R2)+             ;
024662  112722  000002          MOVB    #RMWC,(R2)+             ;
```

```
6668  024666  112722  000000                  MOVB    #RMCS1,(R2)+            ;
6669  024672  112722  000200                  MOVB    #200,(R2)+              ;
6670  024676  004737  044006                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6671  024702  000404                          BR      365$    ;GO TO 365$ IF NO ERROR
6672  024704  000240                          NOP                     ;RETURN HERE IF ERROR
6673  024706  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
6674  024710  000137  024724                  JMP     370$    ;GO TO 370$ IF ERROR WAS FOUND
6675  024714  000240              365$:       NOP
6676  024716  004737  044346                  JSR     PC,TIMOUT               ;WAIT FOR FINISH
6677  024722  000240                          NOP
6678  024724  012737  023074  001122  370$:   MOV     #10$,$LPADR             ;CHANGE LOOP TO START OF TEST
6679  024732  012737  023074  001124          MOV     #10$,$LPERR
6680                                  ;;*****************************************************************
6681                                  ;*TEST 17        WRITE, READ W/ HCI
6682                                  ;;*****************************************************************
6683  024740                          TST17:
6684  024740  000004                          SCOPE                   ;SCOPE CALL
6685  024742  000240                          NOP                     ;START OF TEST
6686  024744  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6687  024750  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6688  024754  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6689  024760  012737  000017  001226          MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6690
6691                                  ;;*****************************************************************
6692                                  ;LOOP #1        FORMAT,WRITE,READ
6693
6694  024766  012704  000000          10$:    MOV     #0,R4                   ;HEADER WORD
6695  024772  012703  000001                  MOV     #1,R3                   ;HCE BIT
6696
6697  024776                          15$:
6698                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
6699  024776  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6700  025002  054130                          .WORD   054130  ;TASK DESCRIPTOR
6701  025004  000404                          BR      20$                     ;GO TO 20$ IF NO ERROR
6702  025006  000240                          NOP                     ;RETURN HERE IF ERROR
6703  025010  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6704  025012  000137  026400                  JMP     370$                    ;GO TO 370$ IF ERROR WAS FOUND
6705  025016                          20$:
6706
6707                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
6708  025016  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
6709  025024  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
6710  025032  012737  107550  001404          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
6711  025040  012737  177376  001402          MOV     #<!C<2+256.>+1>,RMWC0   ;WORD COUNT
6712  025046  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
6713  025054  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
6714
6715                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
6716
6717  025062  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
6718  025066  000405                          BR      25$                     ;GO TO 25$ IF NO ERROR
6719  025070  104401  067466                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
6720  025074  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
6721  025076  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6722  025102                          25$:
6723  025102  012737  071046  001174          MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
```

```
6724  025110  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
6725  025116  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
6726
6727                                  ;CHANGE HEADER WORD TO FORCE ERROR DURING WRITE
6728  025122  030364  107550                  BIT     R3,BUFONE(R4)           ;SET OR RESET BIT??
6729  025126  001403                          BEQ     27$
6730  025130  040364  107550                  BIC     R3,BUFONE(R4)           ;RESET BIT
6731  025134  000402                          BR      28$
6732  025136  050364  107550          27$:    BIS     R3,BUFONE(R4)           ;SET BIT
6733  025142                          28$:
6734                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
6735  025142  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
6736  025146  112722  000034                  MOVB    #RMDC,(R2)+
6737  025152  112722  000006                  MOVB    #RMDA,(R2)+
6738  025156  112722  000004                  MOVB    #RMBA,(R2)+
6739  025162  112722  000002                  MOVB    #RMWC,(R2)+
6740  025166  112722  000032                  MOVB    #RMOF,(R2)+
6741  025172  112722  000000                  MOVB    #RMCS1,(R2)+
6742  025176  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
6743  025202                          30$:
6744
6745                                  ;FORMAT THE DRIVE
6746  025202  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6747  025206  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
6748  025210  000240                          NOP                     ;RETURN HERE IF ERROR
6749  025212  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6750  025214  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6751  025220                          40$:
6752
6753                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
6754  025220  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
6755
6756                                  ;WAIT FOR THE FORMAT TO COMPLETE
6757  025224  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
6758
6759                                  ;GO READ STATUS FOR FORMAT OPERATION
6760  025230  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6761  025234  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
6762  025236  000240                          NOP                     ;RETURN HERE IF ERROR
6763  025240  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6764  025242  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6765  025246                          50$:
6766
6767                                  ;VERIFY NO ERRORS DURING FORMAT
6768  025246  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6769  025252  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
6770  025254  000240                          NOP                     ;RETURN HERE IF ERROR
6771  025256  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6772  025260  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6773  025262  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6774  025266                          60$:
6775
6776                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
6777  025266  012737  025304  001122          MOV     #70$,$LPADR
6778  025274  012737  025304  001124          MOV     #70$,$LPERR
6779  025302  000410                          BR      80$                     ;SKIP TO WRITE OPERATION
```

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

025304                          70$:

                                ;PREPARE DEVICE FOR WRITE OPERATION
025304  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
025310  054130                          .WORD   054130  ;TASK DESCRIPTOR
025312  000404                          BR      80$                     ;GO TO 80$ IF NO ERROR
025314  000240                          NOP                     ;RETURN HERE IF ERROR
025316  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
025320  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
025324                          80$:

025324                          120$:

                                ;WRITE DATA TO THE DRIVE
025324  012737  012000  001432          MOV     #HCI!FMT16,RMOFO        ;INHIBIT HEADER COMPARE
025332  012737  000061  001400          MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
025340  012737  107554  001404          MOV     #BUFONE+4,RMBAO         ;LOAD STARTING BUFFER ADDRESS
025346  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
025352  112722  000006                  MOVB    #RMDA,(R2)+
025356  112722  000034                  MOVB    #RMDC,(R2)+
025362  112722  000032                  MOVB    #RMOF,(R2)+
025366  112722  000004                  MOVB    #RMBA,(R2)+
025372  112722  000002                  MOVB    #RMWC,(R2)+
025376  112722  000000                  MOVB    #RMCS1,(R2)+
025402  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE

025406  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
025412  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
025414  000240                          NOP                     ;RETURN HERE IF ERROR
025416  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
025420  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
025424                          130$:

                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
025424  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR WRITE COMMAND TO COMPLETE
025430  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR WRITE COMMAND
025434  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
025440  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
025442  000240                          NOP                     ;RETURN HERE IF ERROR
025444  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
025446  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
025452                          140$:

                                ;CHECK FOR ERRORS DURING WRITE OPERATION
025452  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
025456  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
```

```
6836  025460  000240                          NOP                             ;RETURN HERE IF ERROR
6837  025462  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
6838  025464  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6839  025466  000137  026214                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
6840  025472                          150$:
6841  025472  032737  000220  001344          BIT     #HCE!FER,RMER1I                 ;ANY ERROR??
6842  025500  001407                          BEQ     151$
6843  025502  013737  001344  001142          MOV     RMER1I,$BDDAT                   ;RECEIVED STATUS
6844  025510  042737  177557  001142          BIC     #^C<HCE!FER>,$BDDAT             ;CLEAR DONT CARES
6845  025516  000412                          BR      152$
6846  025520  032737  100000  001372  151$:   BIT     #BSE,RMER2I                     ;ANY BAD SECTOR ERRROR ??
6847  025526  001414                          BEQ     155$                            ;NO !!
6848  025530  013737  001372  001142          MOV     RMER2I,$BDDAT                   ;RECEIVED STATUS
6849  025536  042737  077777  001142          BIC     #^CBSE,$BDDAT                   ;CLEAR DONT CARES
6850  025544  012737  000000  001140  152$:   MOV     #0,$GDDAT                       ;EXPECTED STATUS
6851  025552  104346                          ERROR   346                             ;HCE W/HCI SET
6852  025554  000137  026214                  JMP     350$
6853  025560                          155$:
6854  025560  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
6855  025564  000405                          BR      160$                            ;GO TO 160$ IF NO ERROR
6856  025566  000240                          NOP                             ;RETURN HERE IF ERROR
6857  025570  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
6858  025572  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6859  025574  000137  026214                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
6860  025600                          160$:
6861
6862  025600                          170$:
6863  025600  004737  045364                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
6864  025604  000405                          BR      180$                            ;GO TO 180$ IF NO ERROR
6865  025606  000240                          NOP                             ;RETURN HERE IF ERROR
6866  025610  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
6867  025612  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
6868  025614  000137  026214                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
6869  025620                          180$:
6870
6871                                  ;CHANGE LOOP ADDRESSES
6872  025620  012737  025634  001122          MOV     #190$,$LPADR
6873  025626  012737  025634  001124          MOV     #190$,$LPERR
6874
6875                                  ;;*****************************************************************
6876                                  ;LOOP #3        READ
6877
6878  025634                          190$:
6879
6880                                  ;PREPARE DEVICE FOR READ OPERATION
6881  025634  004737  040020                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
6882  025640  054130                          .WORD   054130  ;TASK DESCRIPTOR
6883  025642  000404                          BR      200$                            ;GO TO 200$ IF NO ERROR
6884  025644  000240                          NOP                             ;RETURN HERE IF ERROR
6885  025646  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6886  025650  000137  026214                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
6887  025654                          200$:
6888
6889  025654                          240$:
6890
6891                                  ;READ DATA FROM DEVICE
```

```
6892  025654  012737  000071  001400          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
6893  025662  012737  110560  001404          MOV     #BUFTWO+4,RMBA0         ;LOAD STARTING BUFFER ADDRESS
6894  025670  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
6895  025674  112722  000006                  MOVB    #RMDA,(R2)+
6896  025700  112722  000032                  MOVB    #RMOF,(R2)+
6897  025704  112722  000034                  MOVB    #RMDC,(R2)+
6898  025710  112722  000004                  MOVB    #RMBA,(R2)+
6899  025714  112722  000002                  MOVB    #RMWC,(R2)+
6900  025720  112722  000000                  MOVB    #RMCS1,(R2)+
6901  025724  112712  000200                  MOVB    #200,(R2)
6902
6903  025730  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
6904  025734  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
6905  025736  000240                          NOP                     ;RETURN HERE IF ERROR
6906  025740  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
6907  025742  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6908  025746                          250$:
6909
6910                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
6911  025746  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
6912
6913                                  ;WAIT FOR READ OPERATION TO COMPLETE
6914  025752  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
6915
6916                                  ;GO READ STATUS FOR READ OPERATION
6917  025756  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
6918  025762  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
6919  025764  000240                          NOP                     ;RETURN HERE IF ERROR
6920  025766  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
6921  025770  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6922  025774                          260$:
6923
6924                                  ;CHECK FOR ERRORS DURING READ OPERATION
6925  025774  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6926  026000  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
6927  026002  000240                          NOP                     ;RETURN HERE IF ERROR
6928  026004  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6929  026006  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6930  026010  000137  026214                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
6931  026014                          270$:
6932  026014  032737  000220  001344          BIT     #HCE!FER,RMER1I         ;ANY ERROR??
6933  026022  001407                          BEQ     271$                    ;NO!!
6934  026024  013737  001344  001142          MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
6935  026032  042737  177557  001142          BIC     #^C<HCE!FER>,$BDDAT     ;CLEAR DONT CARES
6936  026040  000412                          BR      272$
6937  026042  032737  100000  001372  271$:   BIT     #BSE,RMER2I             ;ANY BAD SECTOR ERROR ??
6938  026050  001414                          BEQ     275$                    ;NO !!
6939  026052  013737  001372  001142          MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
6940  026060  042737  077777  001142          BIC     #^CBSE,$BDDAT           ;CLEAR DONT CARES
6941  026066  012737  000000  001140  272$:   MOV     #0,$GDDAT               ;EXPECTED STATUS
6942  026074  104346                          ERROR   346                     ;HCE W/HCI SET
6943  026076  000137  026214                  JMP     350$
6944  026102                          275$:
6945  026102  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6946  026106  000405                          BR      280$                    ;GO TO 280$ IF NO ERROR
6947  026110  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
                                        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$:

                                                ASL     R3                      ;SHIFT HCE BIT
                                                BNE     320$                    ;CONTINUE IF NOT DONE
                                                TST     R4                      ;SECOND HEADER DONE??
                                                BNE     355$                    ;YES!!
                                                MOV     #1,R3                   ;START WITH BIT 0
                                                MOV     #2,R4                   ;DO SECOND HEADER WORD
                                        320$:
                                                JMP     15$
                                                NOP
                                                NOP
                                        350$:
                                                BR      370$            ;EXIT IF ERROR
                                        355$:
                                                MOV     ASNDC,RMDC0             ;
                                                MOV     ASNDA,RMDA0             ;
                                                MOV     #BUFONE,RMBA0           ;BUFFER ADDRESS
                                                MOV     #-2,RMWC0               ;WORD COUNT
                                                MOV     #FMT16,RMOF0            ;IN 16 BIT MODE
                                                MOV     #WH,RMCS10              ;FORMAT COMMAND
                                                JSR     PC,GENBUF               ;SET UP BUFFER
                                                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                                                .WORD   054130  ;TASK DESCRIPTOR
                                                BR      360$                    ;GO TO 360$ IF NO ERROR
                                                NOP                     ;RETURN HERE IF ERROR
                                                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                                JMP     360$                    ;GO TO 360$ IF ERROR WAS FOUND
                                        360$:
                                                MOV     #WH!GO,RMCS10           ;
                                                MOV     #PUTINX,R2              ;TABLE ADDRESS
                                                MOVB    #RMDA,(R2)+             ;
                                                MOVB    #RMOF,(R2)+             ;
                                                MOVB    #RMDC,(R2)+             ;
                                                MOVB    #RMBA,(R2)+             ;
                                                MOVB    #RMWC,(R2)+             ;
```

```
7004  026344  112722  000000                  MOVB    #RMCS1,(R2)+            ;
7005  026350  112722  000200                  MOVB    #200,(R2)+              ;
7006  026354  004737  044006                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
7007  026360  000404                          BR      365$    ;GO TO 365$ IF NO ERROR
7008  026362  000240                          NOP                             ;RETURN HERE IF ERROR
7009  026364  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
7010  026366  000137  026400                  JMP     370$    ;GO TO 370$ IF ERROR WAS FOUND
7011  026372  000240              365$:       NOP
7012  026374  004737  044346                  JSR     PC,TIMOUT               ;
7013  026400  012737  024766  001122  370$:   MOV     #10$,$LPADR             ;CHANGE LOOP TO START OF TEST
7014  026406  012737  024766  001124          MOV     #10$,$LPERR
7015                                  ;;**************************************************************
7016                                  ;*TEST 20        WRITE, READ W/ IVC ERROR
7017                                  ;;**************************************************************
7018  026414                          TST20:
7019  026414  000004                          SCOPE                           ;SCOPE CALL
7020  026416  000240                          NOP                             ;START OF TEST
7021  026420  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
7022  026424  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
7023  026430  013701  001450                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
7024  026434  012737  000020  001226          MOV     #20,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
7025
7026                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
7027  026442  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = *
7028  026450  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = *, SECTOR = *
7029  026456  012737  107550  001404          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
7030  026464  012737  177522  001402          MOV     #^C256+1>,RMWC0         ;WORD COUNT
7031  026472  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7032  026500  012737  000060  001400          MOV     #WD,RMCS10              ;WRITE HEADER AND DATA COMMAND
7033
7034                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
7035  026506  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
7036
7037
7038                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
7039  026512  012737  026530  001122          MOV     #70$,$LPADR
7040  026520  012737  026530  001124          MOV     #70$,$LPERR
7041  026526  000410                          BR      80$                             ;SKIP TO WRITE OPERATION
7042
7043                                  ;;**************************************************************
7044                                  ;LOOP #2        WRITE,READ
7045
7046  026530                          70$:
7047
7048
7049                                  ;PREPARE DEVICE FOR WRITE OPERATION
7050  026530  004737  040020                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
7051  026534  054130                          .WORD   054130  ;TASK DESCRIPTOR
7052  026536  000404                          BR      80$                             ;GO TO 80$ IF NO ERROR
7053  026540  000240                          NOP                             ;RETURN HERE IF ERROR
7054  026542  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7055  026544  000137  027506                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
7056  026550                          80$:
7057
7058                                  ;RESET VOLUME VALID
7059  026550  112737  000024  001535          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
```

```
7060  026556  112737  000200  001536          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7061  026564  012737  000001  001424          MOV     #DMD,RMMR1O     ;RMMR1=DMD
7062  026572  004737  044006                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
7063  026576  000404                          BR      90$             ;GO TO 90$ IF NO ERROR
7064  026600  000240                          NOP                     ;RETURN HERE IF ERROR
7065  026602  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
7066  026604  000137  027506                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7067  026610                          90$:
7068  026610  112737  000024  001535          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
7069  026616  112737  000200  001536          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7070  026624  012737  000000  001424          MOV     #0,RMMR1O       ;RMMR1=0
7071  026632  004737  044006                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
7072  026636  000404                          BR      100$            ;GO TO 100$ IF NO ERROR
7073  026640  000240                          NOP                     ;RETURN HERE IF ERROR
7074  026642  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
7075  026644  000137  027506                  JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7076  026650                          100$:
7077
7078  026650                          120$:
7079
7080                                  ;WRITE DATA TO THE DRIVE
7081  026650  012737  000061  001400          MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
7082  026656  012737  107554  001404          MOV     #BUFONE+4,RMBA0         ;LOAD STARTING BUFFER ADDRESS
7083  026664  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7084  026670  112722  000006                  MOVB    #RMDA,(R2)+
7085  026674  112722  000034                  MOVB    #RMDC,(R2)+
7086  026700  112722  000032                  MOVB    #RMOF,(R2)+
7087  026704  112722  000004                  MOVB    #RMBA,(R2)+
7088  026710  112722  000002                  MOVB    #RMWC,(R2)+
7089  026714  112722  000000                  MOVB    #RMCS1,(R2)+
7090  026720  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
7091
7092  026724  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7093  026730  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
7094  026732  000240                          NOP                     ;RETURN HERE IF ERROR
7095  026734  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7096  026736  000137  027506                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7097  026742                          130$:
7098
7099                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7100  026742  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7101
7102                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
7103  026746  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7104
7105                                  ;GO READ STATUS FOR WRITE COMMAND
7106  026752  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7107  026756  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
7108  026760  000240                          NOP                     ;RETURN HERE IF ERROR
7109  026762  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7110  026764  000137  027506                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7111  026770                          140$:
7112
7113                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
7114  026770  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7115  026774  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
```

```
7116  026776  000240                          NOP                     ;RETURN HERE IF ERROR
7117  027000  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7118  027002  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7119  027004  000137  027506                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7120  027010                          150$:
7121  027010  032737  010000  001372          BIT     #IVC,RMER2I             ;WAS "IVC" DETECTED??
7122  027016  001024                          BNE     170$                    ;YES!!
7123  027020  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7124  027024  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
7125  027026  000240                          NOP                     ;RETURN HERE IF ERROR
7126  027030  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7127  027032  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7128  027034  000137  027506                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7129  027040                          160$:
7130
7131  027040  013737  001372  001142          MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
7132  027046  013737  001372  001140          MOV     RMER2I,$GDDAT           ;EXPECTED STATUS
7133  027054  052737  010000  001140          BIS     #IVC,$GDDAT
7134  027062  104342                          ERROR   342                     ;IVC NOT DETECTED
7135  027064  000137  027506                  JMP     350$
7136  027070                          170$:
7137  027070  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7138  027074  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
7139  027076  000240                          NOP                     ;RETURN HERE IF ERROR
7140  027100  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7141  027102  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7142  027104  000137  027506                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7143  027110                          180$:
7144
7145                                  ;CHANGE LOOP ADDRESSES
7146  027110  012737  027126  001122          MOV     #190$,$LPADR
7147  027116  012737  027126  001124          MOV     #190$,$LPERR
7148  027124  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
7149
7150                                  ;;*****************************************************************
7151                                  ;LOOP #3        READ
7152
7153  027126                          190$:
7154
7155                                  ;PREPARE DEVICE FOR READ OPERATION
7156  027126  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7157  027132  054130                          .WORD   054130  ;TASK DESCRIPTOR
7158  027134  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
7159  027136  000240                          NOP                     ;RETURN HERE IF ERROR
7160  027140  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7161  027142  000137  027506                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7162  027146                          200$:
7163
7164                                  ;RESET VOLUME VALID
7165  027146  112737  000024  001535          MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
7166  027154  112737  000200  001536          MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7167  027162  012737  000001  001424          MOV     #DMD,RMMR10     ;RMMR1=DMD
7168  027170  004737  044006                  JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
7169  027174  000404                          BR      210$            ;GO TO 210$ IF NO ERROR
7170  027176  000240                          NOP                     ;RETURN HERE IF ERROR
7171  027200  104000                          ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
```

```
7172  027202  000137  027506                    JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7173  027206                            210$:
7174  027206  112737  000024  001535            MOVB    #RMMR1,PUTINX   ;SETUP PUT INDEX TABLE
7175  027214  112737  000200  001536            MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7176  027222  012737  000000  001424            MOV     #0,RMMR1        ;RMMR1=0
7177  027230  004737  044006                    JSR     PC,PUT          ;GO WRITE RMMR1 VIA SUB
7178  027234  000404                            BR      220$            ;GO TO 220$ IF NO ERROR
7179  027236  000240                            NOP                     ;RETURN HERE IF ERROR
7180  027240  104000                            ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
7181  027242  000137  027506                    JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7182  027246                            220$:
7183  027246                            240$:
7184
7185                                    ;READ DATA FROM DEVICE
7186  027246  012737  000071  001400            MOV     #RD!GO,RMCS1    ;READ DATA COMMAND
7187  027254  012737  110560  001404            MOV     #BUFTWO+4,RMBA  ;LOAD STARTING BUFFER ADDRESS
7188  027262  012702  001535                    MOV     #PUTINX,R2      ;LOAD PUT REGISTER INDEX TABLE
7189  027266  112722  000006                    MOVB    #RMDA,(R2)+
7190  027272  112722  000032                    MOVB    #RMOF,(R2)+
7191  027276  112722  000034                    MOVB    #RMDC,(R2)+
7192  027302  112722  000004                    MOVB    #RMBA,(R2)+
7193  027306  112722  000002                    MOVB    #RMWC,(R2)+
7194  027312  112722  000000                    MOVB    #RMCS1,(R2)+
7195  027316  112712  000200                    MOVB    #200,(R2)
7196
7197  027322  004737  044006                    JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7198  027326  000404                            BR      250$            ;GO TO 250$ IF NO ERROR
7199  027330  000240                            NOP                     ;RETURN HERE IF ERROR
7200  027332  104000                            ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7201  027334  000137  027506                    JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7202  027340                            250$:
7203
7204                                    ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7205  027340  004737  043452                    JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7206
7207                                    ;WAIT FOR READ OPERATION TO COMPLETE
7208  027344  004737  044346                    JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7209
7210                                    ;GO READ STATUS FOR READ OPERATION
7211  027350  004737  043536                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7212  027354  000404                            BR      260$            ;GO TO 260$ IF NO ERROR
7213  027356  000240                            NOP                     ;RETURN HERE IF ERROR
7214  027360  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7215  027362  000137  027506                    JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7216  027366                            260$:
7217
7218                                    ;CHECK FOR ERRORS DURING READ OPERATION
7219  027366  004737  044532                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7220  027372  000405                            BR      270$            ;GO TO 270$ IF NO ERROR
7221  027374  000240                            NOP                     ;RETURN HERE IF ERROR
7222  027376  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7223  027400  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7224  027402  000137  027506                    JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7225  027406                            270$:
7226  027406  032737  010000  001372            BIT     #IVC,RMER2I     ;WAS "IVC" DETECTED??
7227  027414  001024                            BNE     290$            ;YES!!
```

```
                                        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$:
                                        MOV     RMER2I,$GDDAT           ;EXPECTED STATUS
                                        BIS     #IVC,$GDDAT
                                        MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
                                        ERROR   342                     ;IVC NOT DETECTED
                                        JMP     350$

                                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$:
                                        MOV     #70$,$LPADR             ;CHANGE LOOP TO START OF TEST
                                        MOV     #70$,$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
```

```
7284  027636                          70$:
7285
7286
7287                                  ;PREPARE DEVICE FOR WRITE OPERATION
7288  027636  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7289  027642  054130                  .WORD   054130  ;TASK DESCRIPTOR
7290  027644  000404                  BR      80$             ;GO TO 80$ IF NO ERROR
7291  027646  000240                  NOP                     ;RETURN HERE IF ERROR
7292  027650  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7293  027652  000137  030470          JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7294  027656                  80$:
7295
7296                                  ;SET UNSAFE ERROR
7297  027656  112737  000014  001535  MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
7298  027664  112737  000200  001536  MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7299  027672  012737  040000  001414  MOV     #UNS,RMER10     ;RMER1 OUTPUT BUFFER = UNS
7300  027700  004737  044006          JSR     PC,PUT          ;WRITE RMER1 VIA PUT SUB
7301  027704  000404                  BR      90$     ;GO TO 90$ IF NO ERROR
7302  027706  000240                  NOP                     ;RETURN HERE TO REPORT ERROR
7303  027710  104000                  ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
7304  027712  000137  030470          JMP     350$    ;GO TO 350$ IF ERROR WAS FOUND
7305  027716                  90$:
7306
7307  027716                  120$:
7308
7309                                  ;WRITE DATA TO THE DRIVE
7310  027716  012737  000061  001400  MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
7311  027724  012702  001535          MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7312  027730  112722  000006          MOVB    #RMDA,(R2)+
7313  027734  112722  000034          MOVB    #RMDC,(R2)+
7314  027740  112722  000032          MOVB    #RMOF,(R2)+
7315  027744  112722  000004          MOVB    #RMBA,(R2)+
7316  027750  112722  000002          MOVB    #RMWC,(R2)+
7317  027754  112722  000000          MOVB    #RMCS1,(R2)+
7318  027760  112722  000200          MOVB    #200,(R2)+              ;TERMINATE TABLE
7319
7320
7321                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7322  027764  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7323  027770  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7324  027774  000404                  BR      130$            ;GO TO 130$ IF NO ERROR
7325  027776  000240                  NOP                     ;RETURN HERE IF ERROR
7326  030000  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7327  030002  000137  030470          JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7328  030006                  130$:
7329  030006  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7330  030012  000404                  BR      135$            ;GO TO 135$ IF NO ERROR
7331  030014  000240                  NOP                     ;RETURN HERE IF ERROR
7332  030016  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7333  030020  000137  030470          JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7334  030024  032737  020000  001342  135$:  BIT     #PIP,RMDSI              ;WAS COMMAND EXECUTED?
7335  030032  001412                  BEQ     136$
7336  030034  013737  001342  001140  MOV     RMDSI,$GDDAT            ;EXPECTED STATUS
7337  030042  042737  020000  001140  BIC     #PIP,$GDDAT
7338  030050  013737  001342  001142  MOV     RMDSI,$BDDAT            ;RECEIVED STATUS
7339  030056  104347                  ERROR   347                     ;NO ABORT
```

```
7340  030060                          136$:
7341
7342                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
7343  030060  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7344
7345                                  ;GO READ STATUS FOR WRITE COMMAND
7346  030064  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7347  030070  000404                        BR      140$                    ;GO TO 140$ IF NO ERROR
7348  030072  000240                        NOP                     ;RETURN HERE IF ERROR
7349  030074  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7350  030076  000137  030470                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7351  030102                          140$:
7352
7353                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
7354  030102  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7355  030106  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
7356  030110  000240                        NOP                     ;RETURN HERE IF ERROR
7357  030112  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7358  030114  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7359  030116  000137  030470                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7360  030122                          150$:
7361
7362  030122  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7363  030126  000405                        BR      180$                    ;GO TO 180$ IF NO ERROR
7364  030130  000240                        NOP                     ;RETURN HERE IF ERROR
7365  030132  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7366  030134  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7367  030136  000137  030470                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7368  030142                          180$:
7369
7370                                  ;CHANGE LOOP ADDRESSES
7371  030142  012737  030160  001122        MOV     #190$,$LPADR
7372  030150  012737  030160  001124        MOV     #190$,$LPERR
7373  030156  000410                        BR      200$                    ;SKIP TO NEXT OPERATION
7374
7375                                  ;;*****************************************************************
7376                                  ;LOOP #3        READ
7377
7378  030160                          190$:
7379
7380                                  ;PREPARE DEVICE FOR READ OPERATION
7381  030160  004737  040020              JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
7382  030164  054130                      .WORD   054130  ;TASK DESCRIPTOR
7383  030166  000404                      BR      200$                    ;GO TO 200$ IF NO ERROR
7384  030170  000240                      NOP                     ;RETURN HERE IF ERROR
7385  030172  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7386  030174  000137  030470              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7387  030200                          200$:
7388                                  ;SET UNSAFE ERROR
7389  030200  112737  000014  001535        MOVB    #RMER1,PUTINX   ;SETUP PUT INDEX TABLE
7390  030206  112737  000200  001536        MOVB    #200,PUTINX+1   ;WRITE TERMINATOR BYTE
7391  030214  012737  040000  001414        MOV     #UNS,RMER10     ;RMER1 OUTPUT BUFFER = UNS
7392  030222  004737  044006                JSR     PC,PUT          ;WRITE RMER1 VIA PUT SUB
7393  030226  000404                        BR      210$    ;GO TO 210$ IF NO ERROR
7394  030230  000240                        NOP                     ;RETURN HERE TO REPORT ERROR
7395  030232  104000                        ERROR                   ;ERROR NUMBER DEFINED BY PUT SUB
```

```
7396  030234  000137  030470                  JMP     350$    ;GO TO 350$ IF ERROR WAS FOUND
7397  030240                          210$:
7398
7399  030240                          240$:
7400
7401                                  ;READ DATA FROM DEVICE
7402  030240  012737  000071  001400          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
7403  030246  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7404  030252  112722  000006                  MOVB    #RMDA,(R2)+
7405  030256  112722  000032                  MOVB    #RMOF,(R2)+
7406  030262  112722  000034                  MOVB    #RMDC,(R2)+
7407  030266  112722  000004                  MOVB    #RMBA,(R2)+
7408  030272  112722  000002                  MOVB    #RMWC,(R2)+
7409  030276  112722  000000                  MOVB    #RMCS1,(R2)+
7410  030302  112712  000200                  MOVB    #200,(R2)
7411
7412                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7413  030306  004737  043452                  JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7414
7415  030312  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7416  030316  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
7417  030320  000240                          NOP                     ;RETURN HERE IF ERROR
7418  030322  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7419  030324  000137  030470                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7420  030330                          250$:
7421
7422                                  ;SEE IF DEVICE STARTED COMMAND
7423  030330  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7424  030334  000404                          BR      255$                    ;GO TO 255$ IF NO ERROR
7425  030336  000240                          NOP                     ;RETURN HERE IF ERROR
7426  030340  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7427  030342  000137  030470                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7428  030346  032737  020000  001342  255$:   BIT     #PIP,RMDSI              ;WAS COMMAND EXECUTED??
7429  030354  001414                          BEQ     256$                    ;NO!!
7430  030356  013737  001342  001140          MOV     RMDSI,$GDDAT            ;EXPECTED STATUS
7431  030364  042737  020000  001140          BIC     #PIP,$GDDAT
7432  030372  013737  001342  001142          MOV     RMDSI,$BDDAT            ;RECEIVED STATUS
7433  030400  104347                          ERROR   347                     ;NO ABORT
7434  030402  000137  030470                  JMP     350$
7435  030406                          256$:
7436
7437                                  ;WAIT FOR READ OPERATION TO COMPLETE
7438  030406  004737  044346                  JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7439
7440                                  ;GO READ STATUS FOR READ OPERATION
7441  030412  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7442  030416  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
7443  030420  000240                          NOP                     ;RETURN HERE IF ERROR
7444  030422  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7445  030424  000137  030470                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7446  030430                          260$:
7447
7448                                  ;CHECK FOR ERRORS DURING READ OPERATION
7449  030430  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7450  030434  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
7451  030436  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
7452  030440  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7453  030442  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7454  030444  000137  030470                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7455  030450                        270$:
7456  030450  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7457  030454  000405                        BR      300$                    ;GO TO 300$ IF NO ERROR
7458  030456  000240                        NOP                     ;RETURN HERE IF ERROR
7459  030460  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7460  030462  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7461  030464  000137  030470                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7462  030470                        300$:
7463
7464  030470                        350$:
7465                                ;;*****************************************************************
7466                                ;*TEST 22        WRITE, READ W/ BAI
7467                                ;;*****************************************************************
7468  030470                        TST22:
7469  030470  000004                        SCOPE                   ;SCOPE CALL
7470  030472  000240                        NOP                     ;START OF TEST
7471  030474  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7472  030500  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7473  030504  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7474  030510  012737  000022  001226        MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7475
7476                                ;;*****************************************************************
7477                                ;LOOP #1        FORMAT,WRITE,READ
7478
7479  030516                        10$:
7480
7481                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
7482  030516  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7483  030522  054130                        .WORD   054130  ;TASK DESCRIPTOR
7484  030524  000404                        BR      20$                     ;GO TO 20$ IF NO ERROR
7485  030526  000240                        NOP                     ;RETURN HERE IF ERROR
7486  030530  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7487  030532  000137  031546                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7488  030536                        20$:
7489
7490                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
7491  030536  012737  000000  001434        MOV     #0,RMDC0                ;CYLINDER = 0
7492  030544  012737  000000  001406        MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
7493  030552  012737  107550  001404        MOV     #BUFONE,RMBA0           ;BUS ADDRESS
7494  030560  012737  177376  001402        MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
7495  030566  012737  010000  001432        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7496  030574  012737  000063  001400        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
7497
7498                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
7499
7500  030602  004737  040734                JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
7501  030606  000405                        BR      25$                     ;GO TO 25$ IF NO ERROR
7502  030610  104401  067466                TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
7503  030614  104000                        ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
7504  030616  000137  031546                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7505  030622                        25$:
7506  030622  012737  071046  001174        MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
7507  030630  012737  000001  001176        MOV     #1,$TMP1                ;RANGE OF PATTERN
```

```
7508  030636  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
7509
7510                                          ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
7511  030642  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
7512  030646  112722  000034                  MOVB    #RMDC,(R2)+
7513  030652  112722  000006                  MOVB    #RMDA,(R2)+
7514  030656  112722  000004                  MOVB    #RMBA,(R2)+
7515  030662  112722  000002                  MOVB    #RMWC,(R2)+
7516  030666  112722  000032                  MOVB    #RMOF,(R2)+
7517  030672  112722  000000                  MOVB    #RMCS1,(R2)+
7518  030676  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
7519  030702                                30$:
7520
7521                                          ;FORMAT THE DRIVE
7522  030702  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7523  030706  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
7524  030710  000240                          NOP                     ;RETURN HERE IF ERROR
7525  030712  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7526  030714  000137  031546                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7527  030720                                40$:
7528
7529                                          ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
7530  030720  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
7531
7532                                          ;WAIT FOR THE FORMAT TO COMPLETE
7533  030724  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
7534
7535                                          ;GO READ STATUS FOR FORMAT OPERATION
7536  030730  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7537  030734  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
7538  030736  000240                          NOP                     ;RETURN HERE IF ERROR
7539  030740  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7540  030742  000137  031546                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7541  030746                                50$:
7542
7543                                          ;VERIFY NO ERRORS DURING FORMAT
7544  030746  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
7545  030752  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
7546  030754  000240                          NOP                     ;RETURN HERE IF ERROR
7547  030756  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7548  030760  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7549  030762  000137  031546                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7550  030766                                60$:
7551
7552                                          ;MOVE LOOP ADDRESSES TO NEXT OPERATION
7553  030766  012737  031004  001122          MOV     #70$,$LPADR
7554  030774  012737  031004  001124          MOV     #70$,$LPERR
7555  031002  000410                          BR      80$                     ;SKIP TO WRITE OPERATION
7556
7557                                          ;;*****************************************************************
7558                                          ;LOOP #2        WRITE,READ
7559
7560  031004                                70$:
7561
7562
7563                                          ;PREPARE DEVICE FOR WRITE OPERATION
```

```
7564  031004  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7565  031010  054130                  .WORD   054130  ;TASK DESCRIPTOR
7566  031012  000404                  BR      80$                     ;GO TO 80$ IF NO ERROR
7567  031014  000240                  NOP                     ;RETURN HERE IF ERROR
7568  031016  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7569  031020  000137  031546          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7570  031024                  80$:
7571
7572
7573  031024                  120$:
7574
7575                          ;WRITE DATA TO THE DRIVE
7576  031024  111102                  MOVB    (R1),R2                 ;GENERATE CS2
7577  031026  042702  177770          BIC     #^C<U0!U1!U2>,R2
7578  031032  042702  000010          BIC     #BAI,R2
7579  031036  010237  001410          MOV     R2,RMCS20               ;INHIBIT ADDRESS INCREMENT
7580  031042  012737  177400  001402  MOV     #<^C256.+1>,RMWC0       ;CHANGE WORD COUNT
7581  031050  012737  107554  001404  MOV     #BUFONE+4,RMBA0         ;CHANGE ADDRESS
7582  031056  012737  000061  001400  MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
7583  031064  012702  001535          MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7584  031070  112722  000010          MOVB    #RMCS2,(R2)+
7585  031074  112722  000006          MOVB    #RMDA,(R2)+
7586  031100  112722  000034          MOVB    #RMDC,(R2)+
7587  031104  112722  000032          MOVB    #RMOF,(R2)+
7588  031110  112722  000004          MOVB    #RMBA,(R2)+
7589  031114  112722  000002          MOVB    #RMWC,(R2)+
7590  031120  112722  000000          MOVB    #RMCS1,(R2)+
7591  031124  112722  000200          MOVB    #200,(R2)+              ;TERMINATE TABLE
7592
7593  031130  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7594  031134  000404                  BR      130$                    ;GO TO 130$ IF NO ERROR
7595  031136  000240                  NOP                     ;RETURN HERE IF ERROR
7596  031140  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7597  031142  000137  031546          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7598  031146                  130$:
7599
7600                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7601  031146  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7602
7603                          ;WAIT FOR WRITE COMMAND TO COMPLETE
7604  031152  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7605
7606                          ;GO READ STATUS FOR WRITE COMMAND
7607  031156  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7608  031162  000404                  BR      140$                    ;GO TO 140$ IF NO ERROR
7609  031164  000240                  NOP                     ;RETURN HERE IF ERROR
7610  031166  104000                  ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7611  031170  000137  031546          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7612  031174                  140$:
7613
7614                          ;CHECK FOR ERRORS DURING WRITE OPERATION
7615  031174  004737  044532          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7616  031200  000405                  BR      150$                    ;GO TO 150$ IF NO ERROR
7617  031202  000240                  NOP                     ;RETURN HERE IF ERROR
7618  031204  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7619  031206  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
```

```
7620  031210  000137  031546              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7621  031214                          150$:
7622  031214  004737  057036              JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7623  031220  000405                      BR      160$                    ;GO TO 160$ IF NO ERROR
7624  031222  000240                      NOP                     ;RETURN HERE IF ERROR
7625  031224  104000                      ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7626  031226  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7627  031230  000137  031546              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7628  031234                          160$:
7629
7630  031234                          170$:
7631  031234  004737  045364              JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7632  031240  000405                      BR      180$                    ;GO TO 180$ IF NO ERROR
7633  031242  000240                      NOP                     ;RETURN HERE IF ERROR
7634  031244  104000                      ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7635  031246  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7636  031250  000137  031546              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7637  031254                          180$:
7638
7639                                  ;CHANGE LOOP ADDRESSES
7640  031254  012737  031272  001122      MOV     #190$,$LPADR
7641  031262  012737  031272  001124      MOV     #190$,$LPERR
7642  031270  000410                      BR      200$                    ;SKIP TO NEXT OPERATION
7643
7644                                  ;;*****************************************************************
7645                                  ;LOOP #3        READ
7646
7647  031272                          190$:
7648
7649                                  ;PREPARE DEVICE FOR READ OPERATION
7650  031272  004737  040020              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7651  031276  054130                      .WORD   054130  ;TASK DESCRIPTOR
7652  031300  000404                      BR      200$                    ;GO TO 200$ IF NO ERROR
7653  031302  000240                      NOP                     ;RETURN HERE IF ERROR
7654  031304  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7655  031306  000137  031546              JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7656  031312                          200$:
7657
7658  031312                          240$:
7659
7660                                  ;READ DATA FROM DEVICE
7661  031312  111102                      MOVB    (R1),R2                 ;RESET BAI
7662  031314  042702  177770              BIC     #^C<U0!U1!U2>,R2
7663  031320  010237  001410              MOV     R2,RMCS20
7664  031324  012737  110560  001404      MOV     #BUFTW0+4,RMBA0         ;CHANGE BUFFER
7665  031332  012737  000071  001400      MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
7666  031340  012702  001535              MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7667  031344  112722  000006              MOVB    #RMDA,(R2)+
7668  031350  112722  000032              MOVB    #RMOF,(R2)+
7669  031354  112722  000034              MOVB    #RMDC,(R2)+
7670  031360  112722  000004              MOVB    #RMBA,(R2)+
7671  031364  112722  000002              MOVB    #RMWC,(R2)+
7672  031370  112722  000000              MOVB    #RMCS1,(R2)+
7673  031374  112712  000200              MOVB    #200,(R2)
7674
7675  031400  004737  044006              JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
```

```
7676  031404  000404                  BR      250$                    ;GO TO 250$ IF NO ERROR
7677  031406  000240                  NOP                             ;RETURN HERE IF ERROR
7678  031410  104000                  ERROR                           ;ERROR # DEFINED BY PUT SUBROUTINE
7679  031412  000137  031546          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7680  031416                  250$:
7681
7682                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7683  031416  004737  043452          JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
7684
7685                          ;WAIT FOR READ OPERATION TO COMPLETE
7686  031422  004737  044346          JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
7687
7688                          ;GO READ STATUS FOR READ OPERATION
7689  031426  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7690  031432  000404                  BR      260$                    ;GO TO 260$ IF NO ERROR
7691  031434  000240                  NOP                             ;RETURN HERE IF ERROR
7692  031436  104000                  ERROR                           ;ERROR # DEFINED BY GET SUBROUTINE
7693  031440  000137  031546          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7694  031444                  260$:
7695
7696                          ;CHECK FOR ERRORS DURING READ OPERATION
7697  031444  004737  044532          JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
7698  031450  000405                  BR      270$                    ;GO TO 270$ IF NO ERROR
7699  031452  000240                  NOP                             ;RETURN HERE IF ERROR
7700  031454  104000                  ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
7701  031456  004736                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7702  031460  000137  031546          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7703  031464                  270$:
7704  031464  004737  057036          JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
7705  031470  000405                  BR      280$                    ;GO TO 280$ IF NO ERROR
7706  031472  000240                  NOP                             ;RETURN HERE IF ERROR
7707  031474  104000                  ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
7708  031476  004736                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7709  031500  000137  031546          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7710  031504                  280$:
7711
7712  031504                  290$:
7713  031504  004737  045364          JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
7714  031510  000405                  BR      300$                    ;GO TO 300$ IF NO ERROR
7715  031512  000240                  NOP                             ;RETURN HERE IF ERROR
7716  031514  104000                  ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
7717  031516  004736                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7718  031520  000137  031546          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7719  031524                  300$:
7720  031524  004737  043104          JSR     PC,CMPBUF               ;GO COMPARE WRITE, READ DATA BUFFERS
7721  031530  107554                  .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
7722  031532  110560                  .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
7723  031534  000404                  BR      310$                    ;GO TO 310$ IF NO ERROR
7724  031536  000240                  NOP                             ;RETURN HERE IF ERROR
7725  031540  104000                  ERROR                           ;ERROR # DEFINED BY CMPBUF SUBROUTINE
7726  031542  000137  031546          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7727  031546                  310$:
7728
7729  031546                  350$:
7730                          ;;*********************************************************************
7731                          ;*TEST 23        WRITE, READ EACH CURRENT LEVEL
```

```
                                ;;*****************************************************************
031546                          T$T23:
031546  000004                          SCOPE                   ;SCOPE CALL
031550  000240                          NOP                     ;START OF TEST
031552  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
031556  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
031562  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
031566  012737  000023  001226          MOV     #23,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

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

031574                          10$:

                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
031574  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
031600  054130                  .WORD   054130  ;TASK DESCRIPTOR
031602  000404                  BR      20$                     ;GO TO 20$ IF NO ERROR
031604  000240                  NOP                             ;RETURN HERE IF ERROR
031606  104000                  ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
031610  000137  032620          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
031614                          20$:

                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
031614  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
031622  012737  000000  001406  25$:     MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
031630  012737  107550  001404          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
031636  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
031644  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
031652  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND

                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE

031660  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
031664  000405                  BR      26$                     ;GO TO 26$ IF NO ERROR
031666  104401  067466          TYPE    ,SCTMSG                 ;TYPE BAD SECTOR MESSAGE
031672  104000                  ERROR                           ;ERROR # DEFINED BY BADSCT SUBROUTINE
031674  000137  032620          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
031700                          26$:
031700  012737  071110  001174          MOV     #ZEROS,$TMP0            ;STARTING ADDRESS OF PATTERN
031706  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
031714  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT

                                ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
031720  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
031724  112722  000034                  MOVB    #RMDC,(R2)+
031730  112722  000006                  MOVB    #RMDA,(R2)+
031734  112722  000004                  MOVB    #RMBA,(R2)+
031740  112722  000002                  MOVB    #RMWC,(R2)+
031744  112722  000032                  MOVB    #RMOF,(R2)+
031750  112722  000000                  MOVB    #RMCS1,(R2)+
031754  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
031760                          30$:

                                ;FORMAT THE DRIVE
031760  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
```

```
7788  031764  000404                        BR      40$                     ;GO TO 40$ IF NO ERROR
7789  031766  000240                        NOP                     ;RETURN HERE IF ERROR
7790  031770  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7791  031772  000137  032620                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7792  031776                        40$:
7793
7794                                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
7795  031776  004737  043452                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
7796
7797                                ;WAIT FOR THE FORMAT TO COMPLETE
7798  032002  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
7799
7800                                ;GO READ STATUS FOR FORMAT OPERATION
7801  032006  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7802  032012  000404                        BR      50$                     ;GO TO 50$ IF NO ERROR
7803  032014  000240                        NOP                     ;RETURN HERE IF ERROR
7804  032016  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7805  032020  000137  032620                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7806  032024                        50$:
7807
7808                                ;VERIFY NO ERRORS DURING FORMAT
7809  032024  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7810  032030  000405                        BR      60$                     ;GO TO 60$ IF NO ERROR
7811  032032  000240                        NOP                     ;RETURN HERE IF ERROR
7812  032034  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7813  032036  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7814  032040  000137  032620                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7815  032044                        60$:
7816
7817                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
7818  032044  012737  032062  001122        MOV     #70$,$LPADR
7819  032052  012737  032062  001124        MOV     #70$,$LPERR
7820  032060  000410                        BR      80$                     ;SKIP TO WRITE OPERATION
7821
7822                                ;;*****************************************************************
7823                                ;LOOP #2        WRITE,READ
7824
7825  032062                        70$:
7826
7827
7828                                ;PREPARE DEVICE FOR WRITE OPERATION
7829  032062  004737  040020                JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
7830  032066  054130                        .WORD   054130  ;TASK DESCRIPTOR
7831  032070  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
7832  032072  000240                        NOP                     ;RETURN HERE IF ERROR
7833  032074  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7834  032076  000137  032620                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7835  032102                        80$:
7836
7837
7838  032102                        120$:
7839
7840                                ;WRITE DATA TO THE DRIVE
7841  032102  012737  177400  001402        MOV     #<^C256.+1>,RMWCO       ;CHANGE WORD COUNT
7842  032110  012737  107554  001404        MOV     #BUFONE+4,RMBAO         ;CHANGE ADDRESS
7843  032116  012737  000061  001400        MOV     #WD!GO,RMCS1O           ;WRITE DATA COMMAND
```

```
7844  032124  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7845  032130  112722  000006                  MOVB    #RMDA,(R2)+
7846  032134  112722  000034                  MOVB    #RMDC,(R2)+
7847  032140  112722  000032                  MOVB    #RMOF,(R2)+
7848  032144  112722  000004                  MOVB    #RMBA,(R2)+
7849  032150  112722  000002                  MOVB    #RMWC,(R2)+
7850  032154  112722  000000                  MOVB    #RMCS1,(R2)+
7851  032160  112722  000200                  MOVB    #200,(R2)+              ;TERMINATE TABLE
7852
7853  032164  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7854  032170  000404                          BR      130$                    ;GO TO 130$ IF NO ERROR
7855  032172  000240                          NOP                     ;RETURN HERE IF ERROR
7856  032174  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7857  032176  000137  032620                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7858  032202                          130$:
7859
7860                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7861  032202  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
7862
7863                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
7864  032206  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
7865
7866                                  ;GO READ STATUS FOR WRITE COMMAND
7867  032212  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7868  032216  000404                          BR      140$                    ;GO TO 140$ IF NO ERROR
7869  032220  000240                          NOP                     ;RETURN HERE IF ERROR
7870  032222  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7871  032224  000137  032620                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7872  032230                          140$:
7873
7874                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
7875  032230  004737  044532                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
7876  032234  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
7877  032236  000240                          NOP                     ;RETURN HERE IF ERROR
7878  032240  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7879  032242  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7880  032244  000137  032620                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7881  032250                          150$:
7882  032250  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
7883  032254  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
7884  032256  000240                          NOP                     ;RETURN HERE IF ERROR
7885  032260  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7886  032262  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7887  032264  000137  032620                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7888  032270                          160$:
7889
7890  032270                          170$:
7891  032270  004737  045364                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
7892  032274  000405                          BR      180$                    ;GO TO 180$ IF NO ERROR
7893  032276  000240                          NOP                     ;RETURN HERE IF ERROR
7894  032300  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7895  032302  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
7896  032304  000137  032620                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7897  032310                          180$:
7898
7899                                  ;CHANGE LOOP ADDRESSES
```

```
7900  032310  012737  032326  001122          MOV     #190$,$LPADR
7901  032316  012737  032326  001124          MOV     #190$,$LPERR
7902  032324  000410                          BR      200$                    ;SKIP TO NEXT OPERATION
7903
7904                                  ;;*****************************************************************
7905                                  ;LOOP #3        READ
7906
7907  032326                          190$:
7908
7909                                  ;PREPARE DEVICE FOR READ OPERATION
7910  032326  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7911  032332  054130                          .WORD   054130  ;TASK DESCRIPTOR
7912  032334  000404                          BR      200$                    ;GO TO 200$ IF NO ERROR
7913  032336  000240                          NOP                     ;RETURN HERE IF ERROR
7914  032340  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7915  032342  000137  032620                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7916  032346                          200$:
7917
7918  032346                          240$:
7919
7920                                  ;READ DATA FROM DEVICE
7921  032346  012737  110560  001404          MOV     #BUFTWO+4,RMBAO         ;CHANGE ADDRESS
7922  032354  012737  177400  001402          MOV     #<^C256.+1>,RMWCO
7923  032362  012737  000071  001400          MOV     #RD!GO,RMCS1O           ;READ DATA COMMAND
7924  032370  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
7925  032374  112722  000006                  MOVB    #RMDA,(R2)+
7926  032400  112722  000032                  MOVB    #RMOF,(R2)+
7927  032404  112722  000034                  MOVB    #RMDC,(R2)+
7928  032410  112722  000004                  MOVB    #RMBA,(R2)+
7929  032414  112722  000002                  MOVB    #RMWC,(R2)+
7930  032420  112722  000000                  MOVB    #RMCS1,(R2)+
7931  032424  112712  000200                  MOVB    #200,(R2)
7932
7933  032430  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
7934  032434  000404                          BR      250$                    ;GO TO 250$ IF NO ERROR
7935  032436  000240                          NOP                     ;RETURN HERE IF ERROR
7936  032440  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
7937  032442  000137  032620                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7938  032446                          250$:
7939
7940                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
7941  032446  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
7942
7943                                  ;WAIT FOR READ OPERATION TO COMPLETE
7944  032452  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
7945
7946                                  ;GO READ STATUS FOR READ OPERATION
7947  032456  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
7948  032462  000404                          BR      260$                    ;GO TO 260$ IF NO ERROR
7949  032464  000240                          NOP                     ;RETURN HERE IF ERROR
7950  032466  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
7951  032470  000137  032620                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
7952  032474                          260$:
7953
7954                                  ;CHECK FOR ERRORS D RING READ OPERATION
7955  032474  004737  044532                  JSR     PC  IERR                ;GO CHECK FOR PRIMARY ERRORS
```

```
7956  032500  000405                        BR      270$            ;GO TO 270$ IF NO ERROR
7957  032502  000240                        NOP                     ;RETURN HERE IF ERROR
7958  032504  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7959  032506  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7960  032510  000137  032620                JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7961  032514                        270$:
7962  032514  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7963  032520  000405                        BR      280$            ;GO TO 280$ IF NO ERROR
7964  032522  000240                        NOP                     ;RETURN HERE IF ERROR
7965  032524  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7966  032526  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7967  032530  000137  032620                JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7968  032534                        280$:
7969
7970  032534                        290$:
7971  032534  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7972  032540  000405                        BR      300$            ;GO TO 300$ IF NO ERROR
7973  032542  000240                        NOP                     ;RETURN HERE IF ERROR
7974  032544  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7975  032546  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7976  032550  000137  032620                JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7977  032554                        300$:
7978  032554  004737  043104                JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
7979  032560  107554                        .WORD   BUFONE+4        ;STARTING ADDRESS OF WRITE BUFFER
7980  032562  110560                        .WORD   BUFTWO+4        ;STARTING ADDDRESS OF READ BUFFER
7981  032564  000404                        BR      310$            ;GO TO 310$ IF NO ERROR
7982  032566  000240                        NOP                     ;RETURN HERE IF ERROR
7983  032570  104000                        ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
7984  032572  000137  032620                JMP     350$            ;GO TO 350$ IF ERROR WAS FOUND
7985  032576                        310$:
7986  032576  062737  000200  001434        ADD     #128.,RMDC0     ;ADVANCE TO NEXT THRESHOLD
7987  032604  023727  001434  001400        CMP     RMDC0,#768.     ;DONE??
7988  032612  101002                        BHI     350$            ;YES!!
7989  032614  000137  031622                JMP     25$             ;DO NEXT CURRENT LEVEL
7990
7991  032620                        350$:
7992  032620  012737  031574  001122        MOV     #10$,$LPADR             ;LOOP BACK TO START OF TEST
7993  032626  012737  031574  001124        MOV     #10$,$LPERR
7994                                ;;*****************************************************************
7995                                ;*TEST 24        WRITE, WRITE CHECK W/ CURRENT LEVEL SWITCHING
7996                                ;;*****************************************************************
7997  032634                        TST24:
7998  032634  000004                        SCOPE                   ;SCOPE CALL
7999  032636  000240                        NOP                     ;START OF TEST
8000  032640  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8001  032644  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8002  032650  013701  001450                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8003  032654  012737  000024  001226        MOV     #24,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8004
8005                                ;;*****************************************************************
8006                                ;LOOP #1        FORMAT,WRITE,READ
8007
8008  032662                        10$:
8009
8010                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
8011  032662  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
```

```
8012  032666  054130                        .WORD   054130  ;TASK DESCRIPTOR
8013  032670  000404                        BR      20$                     ;GO TO 20$ IF NO ERROR
8014  032672  000240                        NOP                     ;RETURN HERE IF ERROR
8015  032674  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8016  032676  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8017  032702                        20$:
8018
8019                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
8020  032702  012737  000577  001434        MOV     #383.,RMDC0             ;CYLINDER = 383.
8021  032710  012737  002037  001406        MOV     #2037,RMDA0             ;TRACK = 4, SECTOR = 31
8022  032716  012737  107550  001404        MOV     #BUFONE,RMBA0           ;BUS ADDRESS
8023  032724  012737  176774  001402        MOV     #<^C<2#<256.+2>>+1>,RMWC0               ;WORD COUNT
8024  032732  012737  010000  001432        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
8025  032740  012737  000063  001400        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
8026
8027                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
8028
8029  032746  004737  040734                JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
8030  032752  000405                        BR      25$                     ;GO TO 25$ IF NO ERROR
8031  032754  104401  067466                TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
8032  032760  104000                        ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
8033  032762  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8034  032766                        25$:
8035  032766  012737  071046  001174        MOV     #ONES,$TMP0             ;STARTING ADDRESS OF PATTERN
8036  032774  012737  000001  001176        MOV     #1,$TMP1                ;RANGE OF PATTERN
8037  033002  004737  042640                JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
8038
8039                                ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
8040  033006  012702  001535                MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
8041  033012  112722  000034                MOVB    #RMDC,(R2)+
8042  033016  112722  000006                MOVB    #RMDA,(R2)+
8043  033022  112722  000004                MOVB    #RMBA,(R2)+
8044  033026  112722  000002                MOVB    #RMWC,(R2)+
8045  033032  112722  000032                MOVB    #RMOF,(R2)+
8046  033036  112722  000000                MOVB    #RMCS1,(R2)+
8047  033042  112712  000200                MOVB    #200,(R2)               ;TERMINATE TABLE
8048  033046                        30$:
8049
8050                                ;FORMAT THE DRIVE
8051  033046  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8052  033052  000404                        BR      40$                     ;GO TO 40$ IF NO ERROR
8053  033054  000240                        NOP                     ;RETURN HERE IF ERROR
8054  033056  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8055  033060  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8056  033064                        40$:
8057
8058                                ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
8059  033064  004737  043452                JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
8060
8061                                ;WAIT FOR THE FORMAT TO COMPLETE
8062  033070  004737  044346                JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
8063
8064                                ;GO READ STATUS FOR FORMAT OPERATION
8065  033074  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8066  033100  000404                        BR      50$                     ;GO TO 50$ IF NO ERROR
8067  033102  000240                        NOP                     ;RETURN HERE IF ERROR
```

```
8068  033104  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8069  033106  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8070  033112                        50$:
8071
8072                                ;VERIFY NO ERRORS DURING FORMAT
8073  033112  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8074  033116  000405                        BR      60$                     ;GO TO 60$ IF NO ERROR
8075  033120  000240                        NOP                     ;RETURN HERE IF ERROR
8076  033122  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8077  033124  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8078  033126  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8079  033132                        60$:
8080
8081                                ;MOVE LOOP ADDRESSES TO NEXT OPERATION
8082  033132  012737  033150  001122        MOV     #70$,$LPADR
8083  033140  012737  033150  001124        MOV     #70$,$LPERR
8084  033146  000410                        BR      80$                     ;SKIP TO WRITE OPERATION
8085
8086                                ;;*****************************************************************
8087                                ;LOOP #2        WRITE,READ
8088
8089  033150                        70$:
8090
8091
8092                                ;PREPARE DEVICE FOR WRITE OPERATION
8093  033150  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8094  033154  054130                        .WORD   054130  ;TASK DESCRIPTOR
8095  033156  000404                        BR      80$                     ;GO TO 80$ IF NO ERROR
8096  033160  000240                        NOP                     ;RETURN HERE IF ERROR
8097  033162  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8098  033164  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8099  033170                        80$:
8100
8101
8102  033170                        120$:
8103
8104                                ;WRITE DATA TO THE DRIVE
8105  033170  012737  177000  001402        MOV     #<^C<2*256.>+1>,RMWC0   ;CHANGE WORD COUNT
8106  033176  012737  107554  001404        MOV     #BUFONE+4,RMBA0         ;CHANGE ADDRESS
8107  033204  012737  000061  001400        MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
8108  033212  012702  001535                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8109  033216  112722  000006                MOVB    #RMDA,(R2)+
8110  033222  112722  000034                MOVB    #RMDC,(R2)+
8111  033226  112722  000032                MOVB    #RMOF,(R2)+
8112  033232  112722  000004                MOVB    #RMBA,(R2)+
8113  033236  112722  000002                MOVB    #RMWC,(R2)+
8114  033242  112722  000000                MOVB    #RMCS1,(R2)+
8115  033246  112722  000200                MOVB    #200,(R2)+              ;TERMINATE TABLE
8116
8117  033252  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8118  033256  000404                        BR      130$                    ;GO TO 130$ IF NO ERROR
8119  033260  000240                        NOP                     ;RETURN HERE IF ERROR
8120  033262  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8121  033264  000137  033626                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8122  033270                        130$:
8123
```

```
8124
8125  033270  004737  043452                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8126                                                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8127
8128  033274  004737  044346                ;WAIT FOR WRITE COMMAND TO COMPLETE
8129                                                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8130
8131  033300  004737  043536                ;GO READ STATUS FOR WRITE COMMAND
8132  033304  000404                               JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8133  033306  000240                               BR      140$                    ;GO TO 140$ IF NO ERROR
8134  033310  104000                               NOP                             ;RETURN HERE IF ERROR
8135  033312  000137  033626                       ERROR                           ;ERROR # DEFINED BY GET SUBROUTINE
8136  033316                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8137                                        140$:
8138
8139  033316  004737  044532                ;CHECK FOR ERRORS DURING WRITE OPERATION
8140  033322  000405                               JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
8141  033324  000240                               BR      150$                    ;GO TO 150$ IF NO ERROR
8142  033326  104000                               NOP                             ;RETURN HERE IF ERROR
8143  033330  004736                               ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
8144  033332  000137  033626                       JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8145  033336                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8146  033336  004737  057036                150$:
8147  033342  000405                               JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
8148  033344  000240                               BR      160$                    ;GO TO 160$ IF NO ERROR
8149  033346  104000                               NOP                             ;RETURN HERE IF ERROR
8150  033350  004736                               ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
8151  033352  000137  033626                       JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8152  033356                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8153                                        160$:
8154  033356
8155  033356  004737  045364                170$:
8156  033362  000405                               JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
8157  033364  000240                               BR      180$                    ;GO TO 180$ IF NO ERROR
8158  033366  104000                               NOP                             ;RETURN HERE IF ERROR
8159  033370  004736                               ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
8160  033372  000137  033626                       JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8161  033376                                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8162                                        180$:
8163
8164  033376  012737  033414  001122          ;CHANGE LOOP ADDRESSES
8165  033404  012737  033414  001124                MOV     #190$,$LPADR
8166  033412  000410                               MOV     #190$,$LPERR
8167                                                BR      200$                    ;SKIP TO NEXT OPERATION
8168
8169                                        ;;*****************************************************************
8170                                        ;LOOP #3        READ
8171  033414                                190$:
8172
8173                                        ;PREPARE DEVICE FOR READ OPERATION
8174  033414  004737  040020                       JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
8175  033420  054130                               .WORD   054130  ;TASK DESCRIPTOR
8176  033422  000404                               BR      200$                    ;GO TO 200$ IF NO ERROR
8177  033424  000240                               NOP                             ;RETURN HERE IF ERROR
8178  033426  104000                               ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8179  033430  000137  033626                       JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
```

```
8180  033434                                  200$:
8181
8182  033434                                  240$:
8183
8184                                          ;READ DATA FROM DEVICE
8185  033434  012737  000051  001400                MOV     #WCO!GO,RMCS10          ;READ DATA COMMAND
8186  033442  012702  001535                        MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8187  033446  112722  000006                        MOVB    #RMDA,(R2)+
8188  033452  112722  000032                        MOVB    #RMOF,(R2)+
8189  033456  112722  000034                        MOVB    #RMDC,(R2)+
8190  033462  112722  000004                        MOVB    #RMBA,(R2)+
8191  033466  112722  000002                        MOVB    #RMWC,(R2)+
8192  033472  112722  000000                        MOVB    #RMCS1,(R2)+
8193  033476  112712  000200                        MOVB    #200,(R2)
8194
8195  033502  004737  044006                        JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8196  033506  000404                                 BR      250$                    ;GO TO 250$ IF NO ERROR
8197  033510  000240                                 NOP                     ;RETURN HERE IF ERROR
8198  033512  104000                                 ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8199  033514  000137  033626                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8200  033520                                  250$:
8201
8202                                          ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8203  033520  004737  043452                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8204
8205                                          ;WAIT FOR READ OPERATION TO COMPLETE
8206  033524  004737  044346                        JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8207
8208                                          ;GO READ STATUS FOR READ OPERATION
8209  033530  004737  043536                        JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8210  033534  000404                                 BR      260$                    ;GO TO 260$ IF NO ERROR
8211  033536  000240                                 NOP                     ;RETURN HERE IF ERROR
8212  033540  104000                                 ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8213  033542  000137  033626                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8214  033546                                  260$:
8215
8216                                          ;CHECK FOR ERRORS DURING READ OPERATION
8217  033546  004737  044532                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8218  033552  000405                                 BR      270$                    ;GO TO 270$ IF NO ERROR
8219  033554  000240                                 NOP                     ;RETURN HERE IF ERROR
8220  033556  104000                                 ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8221  033560  004736                                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8222  033562  000137  033626                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8223  033566                                  270$:
8224  033566  004737  057036                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8225  033572  000405                                 BR      280$                    ;GO TO 280$ IF NO ERROR
8226  033574  000240                                 NOP                     ;RETURN HERE IF ERROR
8227  033576  104000                                 ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8228  033600  004736                                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8229  033602  000137  033626                        JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8230  033606                                  280$:
8231
8232  033606                                  290$:
8233  033606  004737  045364                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8234  033612  000405                                 BR      300$                    ;GO TO 300$ IF NO ERROR
8235  033614  000240                                 NOP                     ;RETURN HERE IF ERROR
```

```
033616  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
033620  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
033622  000137  033626                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
033626                          300$:

033626                          350$:
                                ;;*****************************************************************
                                ;*TEST 25        WRITE, READ EARLY PEAK SHIFT
                                ;;*****************************************************************
033626                          T$T25:
033626  000004                          SCOPE                   ;SCOPE CALL
033630  000240                          NOP                     ;START OF TEST
033632  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
033636  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
033642  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
033646  012737  000025  001226          MOV     #25,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

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

033654                          10$:

                                ;PREPARE THE DEVICE FOR FORMAT OPERATION
033654  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
033660  054130                          .WORD   054130  ;TASK DESCRIPTOR
033662  000404                          BR      20$                     ;GO TO 20$ IF NO ERROR
033664  000240                          NOP                     ;RETURN HERE IF ERROR
033666  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
033670  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
033674                          20$:

                                ;LOAD PARAMETERS AND GENERATE DATA BUFFER
033674  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
033702  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
033710  012737  107550  001404          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
033716  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
033724  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
033732  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND

                                ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE

033740  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
033744  000405                          BR      25$                     ;GO TO 25$ IF NO ERROR
033746  104401  067466                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
033752  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
033754  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
033760                          25$:
033760  012737  071322  001174          MOV     #EARLY,$TMP0            ;STARTING ADDRESS OF PATTERN
033766  012737  000001  001176          MOV     #1,$TMP1                ;RANGE OF PATTERN
033774  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT

                                ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
034000  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
034004  112722  000034                  MOVB    #RMDC,(R2)+
034010  112722  000006                  MOVB    #RMDA,(R2)+
034014  112722  000004                  MOVB    #RMBA,(R2)+
```

```
8292  034020  112722  000002          MOVB    #RMWC,(R2)+
8293  034024  112722  000032          MOVB    #RMOF,(R2)+
8294  034030  112722  000000          MOVB    #RMCS1,(R2)+
8295  034034  112712  000200          MOVB    #200,(R2)                       ;TERMINATE TABLE
8296  034040                      30$:
8297
8298                              ;FORMAT THE DRIVE
8299  034040  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8300  034044  000404                  BR      40$                             ;GO TO 40$ IF NO ERROR
8301  034046  000240                  NOP                             ;RETURN HERE IF ERROR
8302  034050  104000                  ERROR                           ;ERROR # DEFINED BY PUT SUBROUTINE
8303  034052  000137  034650          JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
8304  034056                      40$:
8305
8306                              ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
8307  034056  004737  043452          JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
8308
8309                              ;WAIT FOR THE FORMAT TO COMPLETE
8310  034062  004737  044346          JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
8311
8312                              ;GO READ STATUS FOR FORMAT OPERATION
8313  034066  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8314  034072  000404                  BR      50$                             ;GO TO 50$ IF NO ERROR
8315  034074  000240                  NOP                             ;RETURN HERE IF ERROR
8316  034076  104000                  ERROR                           ;ERROR # DEFINED BY GET SUBROUTINE
8317  034100  000137  034650          JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
8318  034104                      50$:
8319
8320                              ;VERIFY NO ERRORS DURING FORMAT
8321  034104  004737  057036          JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
8322  034110  000405                  BR      60$                             ;GO TO 60$ IF NO ERROR
8323  034112  000240                  NOP                             ;RETURN HERE IF ERROR
8324  034114  104000                  ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
8325  034116  004736                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8326  034120  000137  034650          JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
8327  034124                      60$:
8328
8329                              ;MOVE LOOP ADDRESSES TO NEXT OPERATION
8330  034124  012737  034142  001122        MOV     #70$,$LPADR
8331  034132  012737  034142  001124        MOV     #70$,$LPERR
8332  034140  000410                  BR      80$                             ;SKIP TO WRITE OPERATION
8333
8334                              ;;*****************************************************************
8335                              ;LOOP #2        WRITE,READ
8336
8337  034142                      70$:
8338
8339
8340                              ;PREPARE DEVICE FOR WRITE OPERATION
8341  034142  004737  040020          JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
8342  034146  054130                  .WORD   054130  ;TASK DESCRIPTOR
8343  034150  000404                  BR      80$                             ;GO TO 80$ IF NO ERROR
8344  034152  000240                  NOP                             ;RETURN HERE IF ERROR
8345  034154  104000                  ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8346  034156  000137  034650          JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
8347  034162                      80$:
```

```
8348
8349
8350  034162                          120$:
8351
8352                                  ;WRITE DATA TO THE DRIVE
8353  034162  012737  177400  001402          MOV     #<^C256.+1>,RMWCO        ;CHANGE WORD COUNT
8354  034170  012737  107554  001404          MOV     #BUFONE+4,RMBAO          ;CHANGE ADDRESS
8355  034176  012737  000061  001400          MOV     #WD!GO,RMCS1O            ;WRITE DATA COMMAND
8356  034204  012702  001535                  MOV     #PUTINX,R2               ;LOAD PUT REGISTER INDEX TABLE
8357  034210  112722  000006                  MOVB    #RMDA,(R2)+
8358  034214  112722  000034                  MOVB    #RMDC,(R2)+
8359  034220  112722  000032                  MOVB    #RMOF,(R2)+
8360  034224  112722  000004                  MOVB    #RMBA,(R2)+
8361  034230  112722  000002                  MOVB    #RMWC,(R2)+
8362  034234  112722  000000                  MOVB    #RMCS1,(R2)+
8363  034240  112722  000200                  MOVB    #200,(R2)+               ;TERMINATE TABLE
8364
8365  034244  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8366  034250  000404                          BR      130$                     ;GO TO 130$ IF NO ERROR
8367  034252  000240                          NOP                      ;RETURN HERE IF ERROR
8368  034254  104000                          ERROR                    ;ERROR # DEFINED BY PUT SUBROUTINE
8369  034256  000137  034650                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
8370  034262                          130$:
8371
8372                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8373  034262  004737  043452                  JSR     PC,GETSTS        ;GO TO GETSTS SUBROUTINE
8374
8375                                  ;WAIT FOR WRITE COMMAND TO COMPLETE
8376  034266  004737  044346                  JSR     PC,TIMOUT        ;GO TO TIMOUT SUBROUTINE
8377
8378                                  ;GO READ STATUS FOR WRITE COMMAND
8379  034272  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8380  034276  000404                          BR      140$                     ;GO TO 140$ IF NO ERROR
8381  034300  000240                          NOP                      ;RETURN HERE IF ERROR
8382  034302  104000                          ERROR                    ;ERROR # DEFINED BY GET SUBROUTINE
8383  034304  000137  034650                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
8384  034310                          140$:
8385
8386                                  ;CHECK FOR ERRORS DURING WRITE OPERATION
8387  034310  004737  044532                  JSR     PC,PRIERR        ;GO CHECK FOR PRIMARY ERRORS
8388  034314  000405                          BR      150$                     ;GO TO 150$ IF NO ERROR
8389  034316  000240                          NOP                      ;RETURN HERE IF ERROR
8390  034320  104000                          ERROR                    ;ERROR # DEFINED BY PRIERR SUBROUTINE
8391  034322  004736                          JSR     PC,@(SP)+        ;GO BACK FOR MORE ERROR CHECKS
8392  034324  000137  034650                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
8393  034330                          150$:
8394  034330  004737  057036                  JSR     PC,DTASTS        ;GO VERIFY RESULTS OF DATA TRANSFER
8395  034334  000405                          BR      160$                     ;GO TO 160$ IF NO ERROR
8396  034336  000240                          NOP                      ;RETURN HERE IF ERROR
8397  034340  104000                          ERROR                    ;ERROR # DEFINED BY DTASTS SUBROUTINE
8398  034342  004736                          JSR     PC,@(SP)+        ;GO BACK FOR MORE ERROR CHECKS
8399  034344  000137  034650                  JMP     350$                     ;GO TO 350$ IF ERROR WAS FOUND
8400  034350                          160$:
8401
8402  034350                          170$:
8403  034350  004737  045364                  JSR     PC,SECERR        ;GO CHECK FOR SECONDARY ERRORS
```

```
8404  034354  000405                        BR      180$                    ;GO TO 180$ IF NO ERROR
8405  034356  000240                        NOP                     ;RETURN HERE IF ERROR
8406  034360  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8407  034362  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8408  034364  000137  034650                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8409  034370                        180$:
8410
8411                                  ;CHANGE LOOP ADDRESSES
8412  034370  012737  034406  001122        MOV     #190$,$LPADR
8413  034376  012737  034406  001124        MOV     #190$,$LPERR
8414  034404  000410                        BR      200$                    ;SKIP TO NEXT OPERATION
8415
8416                                  ;;*****************************************************************
8417                                  ;LOOP #3        READ
8418
8419  034406                        190$:
8420
8421                                  ;PREPARE DEVICE FOR READ OPERATION
8422  034406  004737  040020                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8423  034412  054130                        .WORD   054130  ;TASK DESCRIPTOR
8424  034414  000404                        BR      200$                    ;GO TO 200$ IF NO ERROR
8425  034416  000240                        NOP                     ;RETURN HERE IF ERROR
8426  034420  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8427  034422  000137  034650                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8428  034426                        200$:
8429
8430  034426                        240$:
8431
8432                                  ;READ DATA FROM DEVICE
8433  034426  012737  000071  001400        MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
8434  034434  012737  110560  001404        MOV     #BUFTWO+4,RMBA0         ;CHANGE BUFFER
8435  034442  012702  001535                MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8436  034446  112722  000006                MOVB    #RMDA,(R2)+
8437  034452  112722  000032                MOVB    #RMOF,(R2)+
8438  034456  112722  000034                MOVB    #RMDC,(R2)+
8439  034462  112722  000004                MOVB    #RMBA,(R2)+
8440  034466  112722  000002                MOVB    #RMWC,(R2)+
8441  034472  112722  000000                MOVB    #RMCS1,(R2)+
8442  034476  112712  000200                MOVB    #200,(R2)
8443
8444  034502  004737  044006                JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8445  034506  000404                        BR      250$                    ;GO TO 250$ IF NO ERROR
8446  034510  000240                        NOP                     ;RETURN HERE IF ERROR
8447  034512  104000                        ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8448  034514  000137  034650                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8449  034520                        250$:
8450
8451                                  ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8452  034520  004737  043452                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8453
8454                                  ;WAIT FOR READ OPERATION TO COMPLETE
8455  034524  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8456
8457                                  ;GO READ STATUS FOR READ OPERATION
8458  034530  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8459  034534  000404                        BR      260$                    ;GO TO 260$ IF NO ERROR
```

```
8460  034536  000240                          NOP                     ;RETURN HERE IF ERROR
8461  034540  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8462  034542  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8463  034546                          260$:
8464
8465                                  ;CHECK FOR ERRORS DURING READ OPERATION
8466  034546  004737  044532                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8467  034552  000405                          BR      270$                    ;GO TO 270$ IF NO ERROR
8468  034554  000240                          NOP                     ;RETURN HERE IF ERROR
8469  034556  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8470  034560  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8471  034562  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8472  034566                          270$:
8473  034566  004737  057036                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8474  034572  000405                          BR      280$                    ;GO TO 280$ IF NO ERROR
8475  034574  000240                          NOP                     ;RETURN HERE IF ERROR
8476  034576  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8477  034600  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8478  034602  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8479  034606                          280$:
8480
8481  034606                          290$:
8482  034606  004737  045364                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8483  034612  000405                          BR      300$                    ;GO TO 300$ IF NO ERROR
8484  034614  000240                          NOP                     ;RETURN HERE IF ERROR
8485  034616  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8486  034620  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8487  034622  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8488  034626                          300$:
8489  034626  004737  043104                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
8490  034632  107554                          .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
8491  034634  110560                          .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
8492  034636  000404                          BR      310$                    ;GO TO 310$ IF NO ERROR
8493  034640  000240                          NOP                     ;RETURN HERE IF ERROR
8494  034642  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
8495  034644  000137  034650                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8496  034650                          310$:
8497
8498  034650                          350$:
8499                                  ;;*****************************************************************
8500                                  ;*TEST 26        WRITE, READ MIXED PEAK SHIFT
8501                                  ;;*****************************************************************
8502  034650                          TST26:
8503  034650  000004                          SCOPE                   ;SCOPE CALL
8504  034652  000240                          NOP                     ;START OF TEST
8505  034654  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8506  034660  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8507  034664  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8508  034670  012737  000026  001226          MOV     #26,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8509
8510                                  ;;*****************************************************************
8511                                  ;LOOP #1        FORMAT,WRITE,READ
8512
8513  034676                          10$:
8514
8515                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
```

```
8516  034676  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8517  034702  054130                  .WORD   054130  ;TASK DESCRIPTOR
8518  034704  000404                  BR      20$                     ;GO TO 20$ IF NO ERROR
8519  034706  000240                  NOP                     ;RETURN HERE IF ERROR
8520  034710  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8521  034712  000137  035672          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8522  034716                  20$:
8523
8524                          ;LOAD PARAMETERS AND GENERATE DATA BUFFER
8525  034716  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
8526  034724  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
8527  034732  012737  107550  001404          MOV     #BUFONE,RMBA0           ;BUS ADDRESS
8528  034740  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
8529  034746  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
8530  034754  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
8531
8532                          ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
8533
8534  034762  004737  040734          JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
8535  034766  000405                  BR      25$                     ;GO TO 25$ IF NO ERROR
8536  034770  104401  067466          TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
8537  034774  104000                  ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
8538  034776  000137  035672          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8539  035002                  25$:
8540  035002  012737  071006  001174          MOV     #MIXED,$TMP0            ;STARTING ADDRESS OF PATTERN
8541  035010  012737  000200  001176          MOV     #128.,$TMP1             ;RANGE OF PATTERN
8542  035016  004737  042640          JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
8543
8544                          ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
8545  035022  012702  001535          MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
8546  035026  112722  000034          MOVB    #RMDC,(R2)+
8547  035032  112722  000006          MOVB    #RMDA,(R2)+
8548  035036  112722  000004          MOVB    #RMBA,(R2)+
8549  035042  112722  000002          MOVB    #RMWC,(R2)+
8550  035046  112722  000032          MOVB    #RMOF,(R2)+
8551  035052  112722  000000          MOVB    #RMCS1,(R2)+
8552  035056  112712  000200          MOVB    #200,(R2)               ;TERMINATE TABLE
8553  035062                  30$:
8554
8555                          ;FORMAT THE DRIVE
8556  035062  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8557  035066  000404                  BR      40$                     ;GO TO 40$ IF NO ERROR
8558  035070  000240                  NOP                     ;RETURN HERE IF ERROR
8559  035072  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8560  035074  000137  035672          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8561  035100                  40$:
8562
8563                          ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
8564  035100  004737  043452          JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8565
8566                          ;WAIT FOR THE FORMAT TO COMPLETE
8567  035104  004737  044346          JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8568
8569                          ;GO READ STATUS FOR FORMAT OPERATION
8570  035110  004737  043536          JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8571  035114  000404                  BR      50$                     ;GO TO 50$ IF NO ERROR
```

```
8572  035116  000240                          NOP                             ;RETURN HERE IF ERROR
8573  035120  104000                          ERROR                           ;ERROR # DEFINED BY GET SUBROUTINE
8574  035122  000137  035672                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
8575  035126                          50$:
8576
8577                                  ;VERIFY NO ERRORS DURING FORMAT
8578  035126  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
8579  035132  000405                          BR      60$                             ;GO TO 60$ IF NO ERROR
8580  035134  000240                          NOP                             ;RETURN HERE IF ERROR
8581  035136  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
8582  035140  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8583  035142  000137  035672                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
8584  035146                          60$:
8585
8586                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
8587  035146  012737  035164  001122          MOV     #70$,$LPADR
8588  035154  012737  035164  001124          MOV     #70$,$LPERR
8589  035162  000410                          BR      80$                             ;SKIP TO WRITE OPERATION
8590
8591                                  ;;*****************************************************************
8592                                  ;LOOP #2        WRITE,READ
8593
8594  035164                          70$:
8595
8596
8597                                  ;PREPARE DEVICE FOR WRITE OPERATION
8598  035164  004737  040020                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
8599  035170  054130                          .WORD   054130  ;TASK DESCRIPTOR
8600  035172  000404                          BR      80$                             ;GO TO 80$ IF NO ERROR
8601  035174  000240                          NOP                             ;RETURN HERE IF ERROR
8602  035176  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8603  035200  000137  035672                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
8604  035204                          80$:
8605
8606
8607  035204                          120$:
8608
8609                                  ;WRITE DATA TO THE DRIVE
8610  035204  012737  177400  001402          MOV     #<-C256.*1>,RMWCO               ;CHANGE WORD COUNT
8611  035212  012737  107554  001404          MOV     #BUFONE+4,RMBAO                 ;CHANGE ADDRESS
8612  035220  012737  000061  001400          MOV     #WD!GO,RMCS10                   ;WRITE DATA COMMAND
8613  035226  012702  001535                  MOV     #PUTINX,R2                      ;LOAD PUT REGISTER INDEX TABLE
8614  035232  112722  000006                  MOVB    #RMDA,(R2)+
8615  035236  112722  000034                  MOVB    #RMDC,(R2)+
8616  035242  112722  000032                  MOVB    #RMOF,(R2)+
8617  035246  112722  000004                  MOVB    #RMBA,(R2)+
8618  035252  112722  000002                  MOVB    #RMWC,(R2)+
8619  035256  112722  000000                  MOVB    #RMCS1,(R2)+
8620  035262  112722  000200                  MOVB    #200,(R2)+                      ;TERMINATE TABLE
8621
8622  035266  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8623  035272  000404                          BR      130$                            ;GO TO 130$ IF NO ERROR
8624  035274  000240                          NOP                             ;RETURN HERE IF ERROR
8625  035276  104000                          ERROR                           ;ERROR # DEFINED BY PUT SUBROUTINE
8626  035300  000137  035672                  JMP     350$                            ;GO TO 350$ IF ERROR WAS FOUND
8627  035304                          130$:
```

```
                                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
035304  004737  043452                JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                ;WAIT FOR WRITE COMMAND TO COMPLETE
035310  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                ;GO READ STATUS FOR WRITE COMMAND
035314  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
035320  000404                        BR      140$                    ;GO TO 140$ IF NO ERROR
035322  000240                        NOP                     ;RETURN HERE IF ERROR
035324  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
035326  000137  035672                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
035332                          140$:

                                ;CHECK FOR ERRORS DURING WRITE OPERATION
035332  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
035336  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
035340  000240                        NOP                     ;RETURN HERE IF ERROR
035342  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
035344  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
035346  000137  035672                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
035352                          150$:
035352  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
035356  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
035360  000240                        NOP                     ;RETURN HERE IF ERROR
035362  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
035364  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
035366  000137  035672                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
035372                          160$:

035372                          170$:
035372  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
035376  000405                        BR      180$                    ;GO TO 180$ IF NO ERROR
035400  000240                        NOP                     ;RETURN HERE IF ERROR
035402  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
035404  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
035406  000137  035672                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
035412                          180$:

                                ;CHANGE LOOP ADDRESSES
035412  012737  035430  001122        MOV     #190$,$LPADR
035420  012737  035430  001124        MOV     #190$,$LPERR
035426  000410                        BR      200$                    ;SKIP TO NEXT OPERATION

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

035430                          190$:

                                ;PREPARE DEVICE FOR READ OPERATION
035430  004737  040020                JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
035434  054130                        .WORD   054130  ;TASK DESCRIPTOR
035436  000404                        BR      200$                    ;GO TO 200$ IF NO ERROR
035440  000240                        NOP                     ;RETURN HERE IF ERROR
035442  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
8684  035444  000137  035672                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8685  035450                            200$:
8686
8687  035450                            240$:
8688
8689                                    ;READ DATA FROM DEVICE
8690  035450  012737  110560  001404            MOV     #BUFTWO+4,RMBA0         ;CHANGE BUFFER
8691  035456  012737  000071  001400            MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
8692  035464  012702  001535                    MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8693  035470  112722  000006                    MOVB    #RMDA,(R2)+
8694  035474  112722  000032                    MOVB    #RMOF,(R2)+
8695  035500  112722  000034                    MOVB    #RMDC,(R2)+
8696  035504  112722  000004                    MOVB    #RMBA,(R2)+
8697  035510  112722  000002                    MOVB    #RMWC,(R2)+
8698  035514  112722  000000                    MOVB    #RMCS1,(R2)+
8699  035520  112712  000200                    MOVB    #200,(R2)
8700
8701  035524  004737  044006                    JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8702  035530  000404                            BR      250$                    ;GO TO 250$ IF NO ERROR
8703  035532  000240                            NOP                     ;RETURN HERE IF ERROR
8704  035534  104000                            ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8705  035536  000137  035672                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8706  035542                            250$:
8707
8708                                    ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
8709  035542  004737  043452                    JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
8710
8711                                    ;WAIT FOR READ OPERATION TO COMPLETE
8712  035546  004737  044346                    JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
8713
8714                                    ;GO READ STATUS FOR READ OPERATION
8715  035552  004737  043536                    JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8716  035556  000404                            BR      260$                    ;GO TO 260$ IF NO ERROR
8717  035560  000240                            NOP                     ;RETURN HERE IF ERROR
8718  035562  104000                            ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8719  035564  000137  035672                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8720  035570                            260$:
8721
8722                                    ;CHECK FOR ERRORS DURING READ OPERATION
8723  035570  004737  044532                    JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
8724  035574  000405                            BR      270$                    ;GO TO 270$ IF NO ERROR
8725  035576  000240                            NOP                     ;RETURN HERE IF ERROR
8726  035600  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8727  035602  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8728  035604  000137  035672                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8729  035610                            270$:
8730  035610  004737  057036                    JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
8731  035614  000405                            BR      280$                    ;GO TO 280$ IF NO ERROR
8732  035616  000240                            NOP                     ;RETURN HERE IF ERROR
8733  035620  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8734  035622  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8735  035624  000137  035672                    JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8736  035630                            280$:
8737
8738  035630                            290$:
8739  035630  004737  045364                    JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
```

```
8740  035634  000405                          BR      300$                    ;GO TO 300$ IF NO ERROR
8741  035636  000240                          NOP                     ;RETURN HERE IF ERROR
8742  035640  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8743  035642  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8744  035644  000137  035672                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8745  035650                          300$:
8746  035650  004737  043104                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
8747  035654  107554                          .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
8748  035656  110560                          .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
8749  035660  000404                          BR      310$                    ;GO TO 310$ IF NO ERROR
8750  035662  000240                          NOP                     ;RETURN HERE IF ERROR
8751  035664  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
8752  035666  000137  035672                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8753  035672                          310$:
8754
8755  035672                          350$:
8756                                  ;;*****************************************************************
8757                                  ;*TEST 27        WRITE, READ EACH TRACK
8758                                  ;;*****************************************************************
8759  035672                          TST27:
8760  035672  000004                          SCOPE                   ;SCOPE CALL
8761  035674  000240                          NOP                     ;START OF TEST
8762  035676  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
8763  035702  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
8764  035706  013701  001450                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
8765  035712  012737  000027  001226          MOV     #27,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
8766
8767                                  ;;*****************************************************************
8768                                  ;LOOP #1        FORMAT,WRITE,READ
8769
8770  035720                          10$:
8771
8772                                  ;PREPARE THE DEVICE FOR FORMAT OPERATION
8773  035720  004737  040020                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8774  035724  054130                          .WORD   054130  ;TASK DESCRIPTOR
8775  035726  000404                          BR      20$                     ;GO TO 20$ IF NO ERROR
8776  035730  000240                          NOP                     ;RETURN HERE IF ERROR
8777  035732  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8778  035734  000137  036736                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8779  035740                          20$:
8780
8781                                  ;LOAD PARAMETERS AND GENERATE DATA BUFFER
8782  035740  012737  000000  001434          MOV     #0,RMDC0                ;CYLINDER = 0
8783  035746  012737  000000  001406          MOV     #0,RMDA0                ;TRACK = 0, SECTOR = 0
8784  035754  012737  107550  001404  25$:    MOV     #BUFONE,RMBA0           ;BUS ADDRESS
8785  035762  012737  177376  001402          MOV     #<^C<2+256.>+1>,RMWC0   ;WORD COUNT
8786  035770  012737  010000  001432          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
8787  035776  012737  000063  001400          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA COMMAND
8788
8789                                  ;VERIFY SECTOR(S) NOT IN BAD SECTOR FILE
8790
8791  036004  004737  040734                  JSR     PC,BADSCT       ;CALL BAD SECTOR MODULE
8792  036010  000405                          BR      26$                     ;GO TO 26$ IF NO ERROR
8793  036012  104401  067466                  TYPE    ,SCTMSG         ;TYPE BAD SECTOR MESSAGE
8794  036016  104000                          ERROR                   ;ERROR # DEFINED BY BADSCT SUBROUTINE
8795  036020  000137  036736                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
```

```
8796  036024                          26$:
8797  036024  012737  071006  001174          MOV     #MIXED,$TMP0            ;STARTING ADDRESS OF PATTERN
8798  036032  012737  000200  001176          MOV     #128.,$TMP1             ;RANGE OF PATTERN
8799  036040  004737  042640                  JSR     PC,GENBUF               ;GO GENERATE BUFFER FOR FORMAT
8800
8801                                  ;LOAD PUT REGISTER INDEX TABLE FOR FORMAT OPERATIION
8802  036044  012702  001535                  MOV     #PUTINX,R2              ;R2 = BYTE ENTRY POSITION
8803  036050  112722  000034                  MOVB    #RMDC,(R2)+
8804  036054  112722  000006                  MOVB    #RMDA,(R2)+
8805  036060  112722  000004                  MOVB    #RMBA,(R2)+
8806  036064  112722  000002                  MOVB    #RMWC,(R2)+
8807  036070  112722  000032                  MOVB    #RMOF,(R2)+
8808  036074  112722  000000                  MOVB    #RMCS1,(R2)+
8809  036100  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
8810  036104                          30$:
8811
8812                                  ;FORMAT THE DRIVE
8813  036104  004737  044006                  JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8814  036110  000404                          BR      40$                     ;GO TO 40$ IF NO ERROR
8815  036112  000240                          NOP                     ;RETURN HERE IF ERROR
8816  036114  104000                          ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8817  036116  000137  036736                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8818  036122                          40$:
8819
8820                                  ;SETUP GET REGISTER INDEX TABLE FOR READING STATUS
8821  036122  004737  043452                  JSR     PC,GETSTS               ;GO TO GETSTS SUBROUTINE
8822
8823                                  ;WAIT FOR THE FORMAT TO COMPLETE
8824  036126  004737  044346                  JSR     PC,TIMOUT               ;GO TO TIMOUT SUBROUTINE
8825
8826                                  ;GO READ STATUS FOR FORMAT OPERATION
8827  036132  004737  043536                  JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8828  036136  000404                          BR      50$                     ;GO TO 50$ IF NO ERROR
8829  036140  000240                          NOP                     ;RETURN HERE IF ERROR
8830  036142  104000                          ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8831  036144  000137  036736                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8832  036150                          50$:
8833
8834                                  ;VERIFY NO ERRORS DURING FORMAT
8835  036150  004737  057036                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
8836  036154  000405                          BR      60$                     ;GO TO 60$ IF NO ERROR
8837  036156  000240                          NOP                     ;RETURN HERE IF ERROR
8838  036160  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8839  036162  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
8840  036164  000137  036736                  JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8841  036170                          60$:
8842
8843                                  ;MOVE LOOP ADDRESSES TO NEXT OPERATION
8844  036170  012737  036206  001122          MOV     #70$,$LPADR
8845  036176  012737  036206  001124          MOV     #70$,$LPERR
8846  036204  000410                          BR      80$                     ;SKIP TO WRITE OPERATION
8847
8848                                  ;;*****************************************************************
8849                                  ;LOOP #2        WRITE,READ
8850
8851  036206                          70$:
```

```
                                        ;PREPARE DEVICE FOR WRITE OPERATION
036206  004737  040020                      JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
036212  054130                              .WORD   054130  ;TASK DESCRIPTOR
036214  000404                              BR      80$                     ;GO TO 80$ IF NO ERROR
036216  000240                              NOP                     ;RETURN HERE IF ERROR
036220  104000                              ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
036222  000137  036736                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
036226                                  80$:

036226                                  120$:

                                        ;WRITE DATA TO THE DRIVE
036226  012737  107554  001404              MOV     #BUFONE+4,RMBA0         ;CHANGE BUS ADDRESS
036234  012737  177400  001402              MOV     #<^C256.+1>,RMWC0       ;CHANGE WORD COUNT
036242  012737  000061  001400              MOV     #WD!GO,RMCS10           ;WRITE DATA COMMAND
036250  012702  001535                      MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
036254  112722  000006                      MOVB    #RMDA,(R2)+
036260  112722  000034                      MOVB    #RMDC,(R2)+
036264  112722  000032                      MOVB    #RMOF,(R2)+
036270  112722  000004                      MOVB    #RMBA,(R2)+
036274  112722  000002                      MOVB    #RMWC,(R2)+
036300  112722  000000                      MOVB    #RMCS1,(R2)+
036304  112722  000200                      MOVB    #200,(R2)+              ;TERMINATE TABLE

036310  004737  044006                      JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
036314  000404                              BR      130$                    ;GO TO 130$ IF NO ERROR
036316  000240                              NOP                     ;RETURN HERE IF ERROR
036320  104000                              ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
036322  000137  036736                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
036326                                  130$:

                                        ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
036326  004737  043452                      JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE

                                        ;WAIT FOR WRITE COMMAND TO COMPLETE
036332  004737  044346                      JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE

                                        ;GO READ STATUS FOR WRITE COMMAND
036336  004737  043536                      JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
036342  000404                              BR      140$                    ;GO TO 140$ IF NO ERROR
036344  000240                              NOP                     ;RETURN HERE IF ERROR
036346  104000                              ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
036350  000137  036736                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
036354                                  140$:

                                        ;CHECK FOR ERRORS DURING WRITE OPERATION
036354  004737  044532                      JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
036360  000405                              BR      150$                    ;GO TO 150$ IF NO ERROR
036362  000240                              NOP                     ;RETURN HERE IF ERROR
036364  104000                              ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
036366  004736                              JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
036370  000137  036736                      JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
036374                                  150$:
```

```
8908  036374  004737  057036          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8909  036400  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
8910  036402  000240                  NOP                     ;RETURN HERE IF ERROR
8911  036404  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8912  036406  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8913  036410  000137  036736          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8914  036414                  160$:
8915
8916  036414                  170$:
8917  036414  004737  045364          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8918  036420  000405                  BR      180$                    ;GO TO 180$ IF NO ERROR
8919  036422  000240                  NOP                     ;RETURN HERE IF ERROR
8920  036424  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
8921  036426  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8922  036430  000137  036736          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8923  036434                  180$:
8924
8925                          ;CHANGE LOOP ADDRESSES
8926  036434  012737  036452  001122          MOV     #190$,$LPADR
8927  036442  012737  036452  001124          MOV     #190$,$LPERR
8928  036450  000410                  BR      200$                    ;SKIP TO NEXT OPERATION
8929
8930                          ;;*****************************************************************
8931                          ;LOOP #3        READ
8932
8933  036452                  190$:
8934
8935                          ;PREPARE DEVICE FOR READ OPERATION
8936  036452  004737  040020          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
8937  036456  054130                  .WORD   054130  ;TASK DESCRIPTOR
8938  036460  000404                  BR      200$                    ;GO TO 200$ IF NO ERROR
8939  036462  000240                  NOP                     ;RETURN HERE IF ERROR
8940  036464  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
8941  036466  000137  036736          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8942  036472                  200$:
8943
8944  036472                  240$:
8945
8946                          ;READ DATA FROM DEVICE
8947  036472  012737  110560  001404          MOV     #BUFTWO+4,RMBA0         ;CHANGE BUS ADDRESS
8948  036500  012737  000071  001400          MOV     #RD!GO,RMCS10           ;READ DATA COMMAND
8949  036506  012702  001535                  MOV     #PUTINX,R2              ;LOAD PUT REGISTER INDEX TABLE
8950  036512  112722  000006                  MOVB    #RMDA,(R2)+
8951  036516  112722  000032                  MOVB    #RMOF,(R2)+
8952  036522  112722  000034                  MOVB    #RMDC,(R2)+
8953  036526  112722  000004                  MOVB    #RMBA,(R2)+
8954  036532  112722  000002                  MOVB    #RMWC,(R2)+
8955  036536  112722  000000                  MOVB    #RMCS1,(R2)+
8956  036542  112712  000200                  MOVB    #200,(R2)
8957
8958  036546  004737  044006          JSR     PC,PUT  ;GO WRITE REGISTERS WITH PUT SUBROUTINE
8959  036552  000404                  BR      250$                    ;GO TO 250$ IF NO ERROR
8960  036554  000240                  NOP                     ;RETURN HERE IF ERROR
8961  036556  104000                  ERROR                   ;ERROR # DEFINED BY PUT SUBROUTINE
8962  036560  000137  036736          JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8963  036564                  250$:
```

```
8964
8965
8966  036564  004737  043452                ;SETUP REGISTER INPUT BUFFER FOR READING STATUS
                        JSR     PC,GETSTS       ;GO TO GETSTS SUBROUTINE
8967
8968                    ;WAIT FOR READ OPERATION TO COMPLETE
8969  036570  004737  044346                JSR     PC,TIMOUT       ;GO TO TIMOUT SUBROUTINE
8970
8971                    ;GO READ STATUS FOR READ OPERATION
8972  036574  004737  043536                JSR     PC,GET  ;GO READ REGISTERS WITH GET SUBROUTINE
8973  036600  000404                        BR      260$                    ;GO TO 260$ IF NO ERROR
8974  036602  000240                        NOP                     ;RETURN HERE IF ERROR
8975  036604  104000                        ERROR                   ;ERROR # DEFINED BY GET SUBROUTINE
8976  036606  000137  036736                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8977  036612                        260$:
8978
8979                    ;CHECK FOR ERRORS DURING READ OPERATION
8980  036612  004737  044532                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
8981  036616  000405                        BR      270$                    ;GO TO 270$ IF NO ERROR
8982  036620  000240                        NOP                     ;RETURN HERE IF ERROR
8983  036622  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
8984  036624  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8985  036626  000137  036736                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8986  036632                        270$:
8987  036632  004737  057036                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
8988  036636  000405                        BR      280$                    ;GO TO 280$ IF NO ERROR
8989  036640  000240                        NOP                     ;RETURN HERE IF ERROR
8990  036642  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
8991  036644  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
8992  036646  000137  036736                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
8993  036652                        280$:
8994
8995  036652                        290$:
8996  036652  004737  045364                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
8997  036656  000405                        BR      300$                    ;GO TO 300$ IF NO ERROR
8998  036660  000240                        NOP                     ;RETURN HERE IF ERROR
8999  036662  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
9000  036664  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
9001  036666  000137  036736                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
9002  036672                        300$:
9003  036672  004737  043104                JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
9004  036676  107554                        .WORD   BUFONE+4                ;STARTING ADDRESS OF WRITE BUFFER
9005  036700  110560                        .WORD   BUFTWO+4                ;STARTING ADDDRESS OF READ BUFFER
9006  036702  000404                        BR      310$                    ;GO TO 310$ IF NO ERROR
9007  036704  000240                        NOP                     ;RETURN HERE IF ERROR
9008  036706  104000                        ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
9009  036710  000137  036736                JMP     350$                    ;GO TO 350$ IF ERROR WAS FOUND
9010  036714                        310$:
9011  036714  062737  000400  001406        ADD     #400,RMDA0              ;ADVANCE TO NEXT TRACK
9012  036722  022737  002037  001406        CMP     #2037,RMDA0             ;DONE??
9013  036730  103402                        BLO     350$                    ;YES!!
9014  036732  000137  035754                JMP     25$                     ;TEST NEXT TRACK
9015
9016  036736                        350$:
9017  036736  012737  035720  001122        MOV     #10$,$LPADR             ;LOOP BACK TO START OF TEST
9018  036744  012737  035720  001124        MOV     #10$,$LPERR
9019                    ;PUT NEWTEST HERE
```

```
 9020  036752                          $EOSP:
 9021  036752  000240                          NOP
 9022  036754  013700  001450                  MOV     TSTQUE,R0
 9023  036760  062700  000002                  ADD     #2,R0
 9024  036764  010037  001450                  MOV     R0,TSTQUE
 9025  036770  005710                          TST     (R0)
 9026  036772  001402                          BEQ     1$
 9027  036774  000137  007100                  JMP     READY
 9028  037000  012737  001452  001450  1$:     MOV     #TSTQUE+2,TSTQUE
 9029                          .SBTTL  END OF PASS ROUTINE
 9030
 9031                          ;;*********************************************************************
 9032                          ;*INCREMENT THE PASS NUMBER ($PASS)
 9033                          ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
 9034                          ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
 9035                          ;*IF THERES A MONITOR GO TO IT
 9036                          ;*IF THERE ISN'T JUMP TO SHUT
 9037
 9038  037006                          $EOP:
 9039  037006  000240                          NOP
 9040  037010  005037  001116                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 9041  037014  005037  001206                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 9042  037020  005237  001230                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
 9043  037024  042737  100000  001230          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 9044  037032  005327                          DEC     (PC)+           ;;LOOP?
 9045  037034  000001                  $EOPCT: .WORD   1
 9046  037036  003063                          BGT     $DOAGN          ;;YES
 9047  037040  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 9048  037042  000001                  $ENDCT: .WORD   1
 9049  037044  037034                          $EOPCT
 9050  037046  104401  037054                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 9051  037052  000407                          BR      64$             ;;GET OVER THE ASCIZ
 9052                          ;;65$:  .ASCIZ  <12><15>/END PASS #/
 9053  037072                  64$:
 9054  037072  013746  001230                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 9055                                                                  ;;TYPE PASS NUMBER
 9056  037076  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 9057  037100  104401  037106                  TYPE    ,67$            ;;TYPE ASCIZ STRING
 9058  037104  000421                          BR      66$             ;;GET OVER THE ASCIZ
 9059                          ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
 9060  037150                  66$:
 9061  037150  013746  001126                  MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
 9062                                                                  ;;TOTAL NUMBER OF ERRORS
 9063  037154  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 9064  037156  104401  001217                  TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
 9065  037162  005037  001126                  CLR     $ERTTL          ;;CLEAR ERROR TOTAL
 9066  037166  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 9067  037172  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 9068  037174  000005                          RESET                   ;;CLEAR THE WORLD
 9069  037176  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 9070  037200  000240                          NOP                     ;;SAVE ROOM
 9071  037202  000240                          NOP                     ;;FOR
 9072  037204  000240                          NOP                     ;;ACT11
 9073  037206                  $DOAGN:
 9074  037206  000137                          JMP     @(PC)+          ;;RETURN
 9075  037210  063404                  $RTNAD: .WORD   SHUT
```

```
 9076  037212      377      377      000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
 9077                037216                       .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.

037216                                  ERRTYP:
037216  104414                                  SAVREG
037220  032777  020000  141726                  BIT     #SW13,@SWR      ;INHIBIT TYPEOUTS??
037226  001402                                  BEQ     1$              ;NO!!
037230  000137  037746                          JMP     21$             ;YES!!
                                        ;TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER
037234  104401  001217                1$:       TYPE    ,$CRLF
037240  104401  037762                          TYPE    ,ERTY00         ;TYPE "UNT#"
037244  013746  001234                          MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
                                                                        ;;TYPE UNIT NUMBER
037250  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
037252     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
037253     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
037254  005037  037752                          CLR     TSTNMB          ;LOAD TEST NUMBER FOR
037260  013737  001226  037752                  MOV     $TESTN,TSTNMB
037266  104401  037770                          TYPE    ,ERTY01         ;TYPE "TST#"
037272  013746  037752                          MOV     TSTNMB,-(SP)    ;;SAVE TSTNMB FOR TYPEOUT
                                                                        ;;TYPE TEST NUMBER
037276  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
037300     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
037301     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
037302  005037  037754                          CLR     ERRNMB          ;LOAD ERROR NUMBER FOR
037306  113737  001130  037754                  MOVB    $ITEMB,ERRNMB   ;TYPEOUT
037314  001406                                  BEQ     2$              ;SKIP IF NO ERROR CALLED
037316  104401  040000                          TYPE    ,ERTY02         ;TYPE "ERR#"
037322  013746  037754                          MOV     ERRNMB,-(SP)    ;;SAVE ERRNMB FOR TYPEOUT
                                                                        ;;TYPE ERROR NUMBER
037326  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
037330     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
037331     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
037332  104401  040007                2$:       TYPE    ,ERTY03         ;TYPE "PC="
037336  013746  001132                          MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;TYPE PROGRAM COUNTER
037342  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
037344     006                                  .BYTE   6               ;;TYPE 6 DIGIT(S)
037345     001                                  .BYTE   1               ;;TYPE LEADING ZEROS
                                        ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
037346  005737  037754                3$:       TST     ERRNMB          ;WAS AN ERROR CALLED?
037352  001575                                  BEQ     21$             ;NO!!
037354  104401  001217                          TYPE    ,$CRLF          ;YES-TYPE CRLF
037360  105037  037760                          CLRB    @OTFLG          ;CLEAR BOT FLAG
```

```
9134  037364  105037  037761                  CLRB    CHRCNT          ;CLEAR CHARACTER COUNTER
9135  037370  013700  037754                  MOV     ERRNMB,R0       ;R0 POINTS TO FIRST OF
9136  037374  006300                          ASL     R0              ;FOUR ENTRIES IN ERROR
9137  037376  006300                          ASL     R0              ;TABLE
9138  037400  006300                          ASL     R0
9139  037402  062700  001554                  ADD     #$ERRTB-8.,R0
9140  037406  011001                          MOV     (R0),R1         ;R1 POINTS TO ERROR MESSAGE
9141                                                                  ;TABLE
9142  037410  001507                          BEQ     13$             ;BRANCH IF NO ERROR MESSAGE
9143                                ;TYPE THE ERROR MESSAGE
9144  037412  012102                  4$:     MOV     (R1)+,R2        ;R2=ADDRESS OF MESSAGE STRING
9145  037414  001505                          BEQ     12$             ;BRANCH IF END OF MESSAGE
9146  037416  010237  037564                  MOV     R2,11$          ;LOAD ADDRESS OF STRING
9147  037422  005037  037756                  CLR     BOTADR          ;CLEAR BOT ADDRESS
9148  037426  112203                  5$:     MOVB    (R2)+,R3        ;END OF STRING??
9149  037430  001454                          BEQ     10$             ;YES!!
9150  037432  122703  000015                  CMPB    #CR,R3          ;CARRIAGE RETURN??
9151  037436  001003                          BNE     6$              ;NO!!
9152  037440  105037  037761                  CLRB    CHRCNT          ;YES-CLEAR CHAR COUNT
9153  037444  000770                          BR      5$              ;GET NEXT CHARACTER
9154  037446  122703  000012          6$:     CMPB    #LF,R3          ;LINE FEED??
9155  037452  001765                          BEQ     5$              ;YES-GET NEXT CHARACTER
9156  037454  122703  000011                  CMPB    #HT,R3          ;HORIZONTAL TAB??
9157  037460  001007                          BNE     8$              ;NO!!
9158  037462  105237  037761          7$:     INCB    CHRCNT          ;ADJUST CHARACTER COUNT
9159  037466  132737  000007  037761          BITB    #7,CHRCNT
9160  037474  001372                          BNE     7$
9161  037476  000407                          BR      9$
9162  037500  105237  037761          8$:     INCB    CHRCNT          ;INCREMENT CHARACTER COUNT
9163  037504  122703  000040                  CMPB    #' ,R3          ;SPACE??
9164  037510  001002                          BNE     9$              ;NO!!
9165  037512  010237  037756                  MOV     R2,BOTADR       ;SAVE ADDRESS OF SPACE
9166  037516  122737  000100  037761  9$:     CMPB    #64.,CHRCNT     ;END OF LINE??
9167  037524  103340                          BHIS    5$              ;NO!!
9168  037526  013704  037756                  MOV     BOTADR,R4       ;GET ADDRESS OF LAST SPACE
9169  037532  001007                          BNE     90$             ;BRANCH IF SPACE DETECTED
9170  037534  104401  001217                  TYPE    ,$CRLF          ;TYPE CRLF
9171  037540  105037  037761                  CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
9172  037544  013702  037564                  MOV     11$,R2          ;SET UP R2 FOR TESTING
9173  037550  000726                          BR      5$
9174  037552  105044                  90$:    CLRB    -(R4)           ;REPLACE SPACE
9175  037554  112737  177777  037760          MOVB    #-1,BOTFLG      ;SET BOT FLAG
9176  037562  104401                  10$:    TYPE                    ;TYPE ERROR MESSAGE STRING
9177  037564  000000                  11$:    .WORD                   ;STRING ADDRESS GOES HERE
9178  037566  105737  037760                  TSTB    BOTFLG          ;WAS STRING TRUNCATED??
9179  037572  001707                          BEQ     4$              ;NO!!
9180  037574  104401  001217                  TYPE    ,$CRLF          ;YES-TYPE CRLF
9181  037600  105037  037760                  CLRB    BOTFLG          ;CLEAR BOT FLAG
9182  037604  105037  037761                  CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
9183  037610  013702  037756                  MOV     BOTADR,R2       ;SETUP R2 FOR TESTING
9184  037614  010237  037564                  MOV     R2,11$          ;SETUP 11$ FOR TYPING
9185  037620  112742  000040                  MOVB    #' ,-(R2)       ;RESTORE SPACE
9186  037624  105722                          TSTB    (R2)+           ;RESTORE R2
9187  037626  000677                          BR      5$              ;TYPE REST OF STRING
9188  037630                          12$:
9189                                ;TYPE ERROR HEADER AND ERROR DATA
```

```
                        13$:
037630                          MOV     2(R0),R1        ;R1 POINTS TO ERROR HEADER TABLE
037630  016001  000002          BEQ     21$             ;BRANCH IF NO HEADER
037634  001444                  TYPE    ,$CRLF          ;(ASSUME NO DATA)
037636  104401  001217          MOV     4(R0),R2        ;R2 POINTS TO DATA ADDRESS TABLE
037642  016002  000004          MOV     6(R0),R3        ;R3 POINTS TO FORMAT TABLE
037646  016003  000006  14$:    MOV     (R1)+,15$       ;PUT HEADER ADDRESS FOR TYPE
037652  012137  037662          BEQ     21$             ;BRANCH IF END OF HEADERS
037656  001433                                          ;(ASSUME END OF DATA)
                                TYPE
037660  104401          15$:    .WORD   0               ;HEADER ADDRESS GOES HERE
037662  000000                  TYPE    ,$CRLF
037664  104401  001217          TST     R2              ;DATA WITH HEADER??
037670  005702                  BEQ     14$             ;NO!!
037672  001767                  MOV     (R2)+,R4        ;R4 POINTS TO DATA ADDRESS
037674  012204                  MOV     (R3)+,R5        ;R5 POINTS TO FORMAT
037676  012305          16$:    TSTB    (R5)+           ;WHAT KIND OF DATA??
037700  105725                  BMI     18$             ;BINARY
037702  100407                  BEQ     17$             ;OCTAL
037704  001403                  MOV     @(R4)+,-(SP)    ;DECIMAL
037706  013446                  TYPDS
037710  104405                  BR      19$
037712  000405          17$:    MOV     @(R4)+,-(SP)
037714  013446                  TYPOC
037716  104402                  BR      19$
037720  000402          18$:    MOV     @(R4)+,-(SP)
037722  013446                  TYPBN
037724  104406          19$:    TST     (R4)            ;MORE DATA??
037726  005714                  BEQ     20$             ;NO!!
037730  001403                  TYPE    ,ERTY04         ;YES-TYPE 2 SPACES
037732  104401  040015          BR      16$             ;AND CONTINUE
037736  000760          20$:    TYPE    ,$CRLF          ;TYPE ONE BLANK LINE
037740  104401  001217          BR      14$             ;BEFORE NEXT HEADER
037744  000742          21$:    RESREG
037746  104415                  RTS     PC
037750  000207

037752  000000          TSTNMB: .WORD   0               ;TEST NUMBER
037754  000000          ERRNMB: .WORD   0               ;ERROR NUMBER
037756  000000          BOTADR: .WORD   0               ;BEGINNING OF TEXT ADDRESS
037760      000         BOTFLG: .BYTE   0               ;BOT FLAG
037761      000         CHRCNT: .BYTE   0               ;CHARACTER COUNT

037762  047125  052111  000043  ERTY00: .ASCIZ  @UNIT#@
037770  020054  042524  052123  ERTY01: .ASCIZ  @, TEST#@
037776  000043
040000  020054  051105  021522  ERTY02: .ASCIZ  @, ERR#@
040006      000
040007      054 050040  036503  ERTY03: .ASCIZ  @, PC=@
040014      000
040015      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

040020                          TSTPRP:

                                ;STORE TASK DESCRIPTOR AND CLEAR USER'S ERROR CALL
040020  017637  000000  040732          MOV     @(SP),500$      ;STORE DESCRIPTOR
040026  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
040032  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR CALL
040036  162716  000004                  SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
                                ;;*****************************************************************
                                ;SELECT DEVICE AND VERIFY DEVICE AVAILABLE IF BIT 15 SET IN TASK
040042  032737  100000  040732          BIT     #BIT15,500$     ;SELECT DEVICE??
040050  001414                          BEQ     30$             ;NO!!
040052  004737  051226                  JSR     PC,DEVSEL       ;GO SELECT DEVICE
040056  000411                          BR      30$             ;NO ERROR - CONTINUE
040060  000401                          BR      20$
040062  000000          10$:    .WORD                           ;ERROR NUMBER FROM DEVSEL
040064  062716  000004  20$:    ADD     #4,(SP)                 ;TRANSFER ERROR TO USER
040070  113776  040062  000000          MOVB    10$,@(SP)
040076  000137  040720                  JMP     400$
040102                          30$:

                                ;;*****************************************************************
                                ;CLEAR CONTROLLER IF BIT 14 IS SET IN TASK
040102  032737  040000  040732          BIT     #BIT14,500$     ;CLEAR CONTROLLER??
040110  001453                          BEQ     120$            ;NO!!
```

```
                                        JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                                        BR      60$             ;CONTINUE - NO ERROR
                                        BR      50$
                                40$:    .WORD                   ;ERROR NUMBER FROM CNTCLR
                                50$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
                                        MOVB    40$,@(SP)
                                        JMP     400$
                                60$:
                                ;;*****************************************************************
                                ;VERIFY CONTROLLER CLEAR IF BIT6 SET IN TASK
                                        BIT     #BIT6,500$      ;VERIFY??
                                        BEQ     120$            ;NO!!
                                        JSR     PC,GETSTS       ;SETUP INDEX TABLE
                                        JSR     PC,GET          ;GO GET STATUS
                                        BR      90$             ;NO ERROR GETTING STATUS
                                        BR      80$
                                70$:    .WORD                   ;ERROR FROM GETTING STATUS
                                80$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
                                        MOVB    70$,@(SP)
                                        JMP     400$
                                90$:    JSR     PC,CLRSTS       ;GO VERIFY STATUS CLEAR
                                        BR      120$            ;NO ERROR IN CLEAR
                                        BR      110$
                                100$:   .WORD                   ;ERROR IN STATUS CLEAR
                                110$:   TST     (SP)+           ;STRIP RETURN ADDRESS TO
                                        ADD     #4,(SP)         ;SUBROUTINE AND TRANSFER
                                        MOVB    100$,@(SP)      ;ERROR TO USER
                                        JMP     400$
                                120$:
                                ;;*****************************************************************
                                ;EXECUTE PACK ACKNOWLEDGE IF BIT12 SET IN TASK AND VOLUME IS
                                ;NOT VALID
                                        BIT     #BIT12,500$     ;PACK ACKNOWLEDGE??
                                        BEQ     240$            ;NO!!
                                        MOVB    #RMDS,GETINX    ;GET RMDS
                                        MOVB    #200,GETINX+1
                                        JSR     PC,GET
                                        BR      150$            ;NO ERROR GETTING RMDS
                                        BR      140$
                                130$:   .WORD
                                140$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
                                        MOVB    130$,@(SP)
                                        JMP     400$
                                150$:   BIT     #VV,RMDSI       ;IS VOLUME VALID??
                                        BNE     240$            ;YES!!
                                        CLR     MEDENB          ;CLEAR MEDIA ENABLE
                                        MOV     #PAKACK!GO,RMCS10       ;LOAD PACK ACK COMMAND
                                        MOVB    #RMCS1,PUTINX   ;SETUP REGISTER INDEX TABLE
                                        MOVB    #200,PUTINX+1
                                        JSR     PC,PUT          ;GO WRITE COMMAND
                                        BR      180$            ;NO ERROR LOADING REGISTER
                                        BR      170$
                                160$:   .WORD                   ;ERROR FROM PUT SUB
                                170$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
                                        MOVB    160$,@(SP)
```

```
                                        BR      400$
                                180$:
                                ;;*****************************************************************
                                ;VERIFY PACK ACKNOWLEDGE IF #BIT4 SET IN TASK
040400  032737  000020  040732          BIT     #BIT4,500$      ;VERIFY PACK ACKNOWLEDGE??
040406  001431                          BEQ     240$            ;NO!!
040410  004737  043452                  JSR     PC,GETSTS       ;SETUP FOR STATUS
040414  004737  043536                  JSR     PC,GET          ;GO GET STATUS
040420  000410                          BR      210$            ;NO ERROR GETTING STATUS
040422  000401                          BR      200$
040424  000000                  190$:   .WORD                   ;ERROR FROM GET SUB
040426  062716  000004          200$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
040432  113776  040424  000000          MOVB    190$,@(SP)
040440  000527                          BR      400$
040442  004737  053676          210$:   JSR     PC,ACKSTS       ;GO CHECK ACKNOWLEDGE
040446  000411                          BR      240$            ;NO ERROR
040450  000401                          BR      230$
040452  000000                  220$:   .WORD                   ;PACK ACKNOWLEDGE ERROR
040454  005726                  230$:   TST     (SP)+           ;STRIP RETURN TO SUB AND
040456  062716  000004                  ADD     #4,(SP)         ;TRANSFER ERROR TO USER
040462  113776  040452  000000          MOVB    220$,@(SP)
040470  000513                          BR      400$
040472                          240$:
                                ;;*****************************************************************
                                ;RECALIBRATE DRIVE IF BIT 11 SET IN TASK AND PIP IS ACTIVE
040472  032737  004000  040732          BIT     #BIT11,500$     ;RECALIBRATE??
040500  001513                          BEQ     410$            ;NO!!
040502  112737  000012  001506          MOVB    #RMDS,GETINX    ;LOAD REGISTER INDEX TABLE
040510  112737  000200  001507          MOVB    #200,GETINX+1
040516  004737  043536                  JSR     PC,GET          ;GO GET RMDS
040522  000410                          BR      270$            ;NO ERROR GETTING RMDS
040524  000401                          BR      260$
040526  000000                  250$:   .WORD                   ;ERROR FROM GET SUB
040530  062716  000004          260$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
040534  113776  040526  000000          MOVB    250$,@(SP)
040542  000466                          BR      400$
040544  032737  020000  001342  270$:   BIT     #PIP,RMDSI      ;IS PIP ACTIVE??
040552  001466                          BEQ     410$            ;NO!!
040554  012737  000007  001400          MOV     #RECAL!GO,RMCS10;LOAD RECALIBRATE COMMAND
040562  112737  000000  001535          MOVB    #RMCS1,PUTINX   ;AND REGISTER INDEX
040570  112737  000200  001536          MOVB    #200,PUTINX+1
040576  004737  044306                  JSR     PC,PUT          ;GO ISSUE RECALIBRATE
040602  000410                          BR      300$            ;NO ERROR
040604  000401                          BR      290$
040606  000000                  280$:   .WORD                   ;ERROR IN REGISTER TRANSFER
040610  062716  000004          290$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
040614  113776  040606  000000          MOVB    280$,@(SP)
040622  000436                          BR      400$
040624  004737  043452          300$:   JSR     PC,GETSTS       ;SETUP FOR STATUS
040630  004737  044346                  JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
                                ;;*****************************************************************
                                ;VERIFY RECALIBRATE IF BIT 3 SET IN TASK
040634  032737  000010  040732          BIT     #BIT3,500$      ;VERIFY RECALIBRATE??
```

```
 9410  040642  001432                        BEQ     410$            ;NO!!
 9411  040644  004737  043536                JSR     PC,GET          ;GO GET STATUS
 9412  040650  000410                        BR      330$            ;NO ERROR GETTING STATUS
 9413  040652  000401                        BR      320$
 9414  040654  000000              310$:     .WORD                   ;ERROR FROM GET
 9415  040656  062716  000004      320$:     ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9416  040662  113776  040654  000000        MOVB    310$,@(SP)
 9417  040670  000413                        BR      400$
 9418  040672  004737  054472              330$:     JSR     PC,RCLSTS       ;GO CHECK RECALIBRATE
 9419  040676  000414                        BR      410$            ;NO ERROR DURING RECALIBRATE
 9420  040700  000401                        BR      350$
 9421  040702  000000              340$:     .WORD                   ;ERROR DURING RECALIBRATE
 9422  040704  005726              350$:     TST     (SP)+           ;STRIP RETURN TO SUB AND
 9423  040706  062716  000004              ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 9424  040712  113776  040702  000000        MOVB    340$,@(SP)
 9425  040720  162716  000002      400$:     SUB     #2,(SP)         ;MOVE SP BACK BEFORE ERROR
 9426  040724  000240                        NOP
 9427  040726  000240                        NOP
 9428  040730  000207              410$:     RTS     PC              ;RETURN TO USER
 9429
 9430  040732  000000              500$:     .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.

040734                  BADSCT:

                        ;TEST "MEDIA ENABLE" TO DETERMINE WHETHER OR NOT THE BAD SECTOR FILES
                        ;HAVE BEEN RECOVERED.
040734  005737  001474          TST     MEDENB
040740  001402                BEQ     10$
040742  000137  042160          JMP     300$                    ;BAD SECTOR FILE IS AVAILABLE

                        ;;*****************************************************************
                        ;RECOVER THE BAD SECTOR FILE FROM THE LAST TRACK
040746                  10$:

                        ;SAVE THE USER'S PUT BUFFER
040746  010046                  MOV     R0,-(SP)                ;;PUSH R0 ON STACK
040750  005000                CLR     R0
040752  016060  001400  107550  20$:    MOV     PUTBUF(R0),BUFFER(R0)
040760  062700  000002          ADD     #2,R0                   ;ADVANCE TO NEXT BUFFER POSITION
040764  022700  000046          CMP     #46,R0                  ;END OF BUFFER
040770  103370                BHIS    20$                     ;NO !!

                        ;SET RETRY COUNT AND LOAD PUT BUFFER AND REGISTER INDEX TABLE
040772  012737  000003  042626          MOV     #3,500$                 ;RETRY COUNT
041000  012737  002000  001406          MOV     #002000,RMDAO           ;STARTING SECTOR ADDRESS
041006  012737  001466  001434          MOV     #822.,RMDCO             ;DESIRED CYLINDER
```

```
 9487  041014  012737  177376  001402          MOV     #<^C258.+1>,RMWCO       ;WORD COUNT
 9488  041022  012737  010000  001432          MOV     #FMT16,RMOFO    ;16 BIT FORMAT
 9489  041030  012737  105530  001404          MOV     #MFGFIL,RMBAO   ;BUFFER ADDRESS
 9490
 9491  041036  012700  001535                  MOV     #PUTINX,R0      ;R0 POINTS TO REGISTER INDEX TABLE
 9492  041042  112720  000006                  MOVB    #RMDA,(R0)+
 9493  041046  112720  000034                  MOVB    #RMDC,(R0)+
 9494  041052  112720  000002                  MOVB    #RMWC,(R0)+
 9495  041056  112720  000032                  MOVB    #RMOF,(R0)+
 9496  041062  112720  000004                  MOVB    #RMBA,(R0)+
 9497  041066  112720  000000                  MOVB    #RMCS1,(R0)+
 9498  041072  112720  000200                  MOVB    #200,(R0)+
 9499  041076  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 9500
 9501                          ;SET GET INDEX TABLE FOR READING STATUS
 9502  041100  004737  043452                  JSR     PC,GETSTS       ;SETUP GET INDEX REGISTER FOR STATUS
 9503  041104                  30$:
 9504
 9505                          ;CLEAR THE DEVICE USING DRIVE CLEAR COMMAND
 9506  041104  012737  000011  001400          MOV     #DRVCLR!GO,RMCS10       ;LOAD COMMAND IN PUT BUFFER
 9507  041112  004737  044006                  JSR     PC,PUT          ;OUTPUT COMMAND
 9508  041116  000411                          BR      45$             ;RETURN HERE IF NO ERROR
 9509  041120  000401                          BR      40$             ;GET AROUND ERROR #
 9510  041122  000000                  35$:    .WORD                   ;ERROR # GOES HERE
 9511
 9512  041124  062716  000006          40$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
 9513  041130  113776  041122  000000          MOVB    35$,@(SP)       ;MOVE ERROR NUMBER TO USER
 9514  041136  000137  041636                  JMP     215$
 9515  041142  004737  043536          45$:    JSR     PC,GET          ;GO GET STATUS
 9516  041146  000411                          BR      60$             ;RETURN HERE IF NO ERROR
 9517  041150  000401                          BR      55$             ;GET AROUND ERROR #
 9518  041152  000000                  50$:    .WORD                   ;ERROR # GOES HERE
 9519
 9520  041154  062716  000006          55$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
 9521  041160  113776  041152  000000          MOVB    50$,@(SP)       ;MOVE ERROR # TO USERS ERROR CALL
 9522  041166  000137  041636                  JMP     215$
 9523
 9524  041172  004737  056234          60$:    JSR     PC,DRVSTS       ;GO VERIFY DRIVE CLEAR COMMAND
 9525  041176  000412                          BR      75$             ;RETURN HERE IF NO ERROR
 9526  041200  000401                          BR      70$             ;GET AROUND ERROR
 9527  041202  000000                  65$:    .WORD                   ;ERROR # GOES HERE
 9528
 9529  041204  005726                  70$:    TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
 9530  041206  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
 9531  041212  113776  041202  000000          MOVB    65$,@(SP)       ;MOVE ERROR # TO USER CALL
 9532  041220  000137  041636                  JMP     215$
 9533
 9534  041224                  75$:
 9535
 9536                          ;ISSUE A PACK ACKNOWLEDGE IF VOLUME VALID IS RESET
 9537  041224  032737  000100  001342          BIT     #VV,RMDSI       ;IS VV RESET ??
 9538  041232  001050                          BNE     120$            ;NO !!
 9539  041234  012737  000023  001400          MOV     #PACACK!GO,RMCS10       ;LOAD COMMAND
 9540  041242  004737  044006                  JSR     PC,PUT          ;GO PUT COMMAND TO DRIVE
 9541  041246  000411                          BR      90$             ;RETURN HERE IF NO OUTPUT ERROR
 9542  041250  000401                          BR      85$             ;GET AROUND ERROR #
```

```
9543  041252  000000                  80$:    .WORD                   ;ERROR # GOES HERE
9544
9545  041254  062716  000006          85$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9546  041260  113776  041252  000000          MOVB    80$,@(SP)       ;MOVE ERROR # TO ERROR CALL
9547  041266  000137  041636                  JMP     215$
9548  041272  004737  043536          90$:    JSR     PC,GET          ;GO GET STATUS FOR PACK ACK
9549  041276  000411                          BR      105$            ;RETURN HERE IF NO ERROR
9550  041300  000401                          BR      100$            ;GET AROUND ERROR #
9551  041302  000000          95$:    .WORD                           ;ERROR # GOES HERE
9552
9553  041304  062716  000006          100$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9554  041310  113776  041302  000000          MOVB    95$,@(SP)       ;MOVE ERROR # TO CALL
9555  041316  000137  041636                  JMP     215$
9556
9557  041322  004737  053676          105$:   JSR     PC,ACKSTS       ;GO VERIFY ACKNOWLEDGE STATUS
9558  041326  000412                          BR      120$            ;RETURN HERE IF NO ERROR
9559  041330  000401                          BR      115$            ;GET AROUND ERROR #
9560  041332  000000          110$:   .WORD                           ;ERROR # GOES HERE
9561
9562  041334  005726          115$:   TST     (SP)+                   ;STRIP RETURN TO SUBROUTINE
9563  041336  062716  000006          ADD     #6,(SP)                 ;MOVE SP TO USERS ERROR CALL
9564  041342  113776  041332  000000  MOVB    110$,@(SP)              ;MOVE ERROR # TO USERS ERROR CALL
9565  041350  000137  041636          JMP     215$
9566
9567  041354                  120$:
9568
9569                  ;RECALIBRATE THE DRIVE IF PIP IS SET
9570  041354  032737  020000  001342        BIT     #PIP,RMDSI      ;IS PIP SET ??
9571  041362  001452                        BEQ     165$            ;NO !!
9572
9573  041364  012737  000007  001400        MOV     #RECAL!GO,RMCS10        ;LOAD RECALIBRATE COMMAND
9574  041372  004737  044006                JSR     PC,PUT          ;PUT THE RECAL COMMAND
9575  041376  000411                        BR      135$            ;RETURN HERE IF NO ERROR
9576  041400  000401                        BR      130$            ;GET AROUND ERROR #
9577  041402  000000          125$:   .WORD                           ;ERROR # GOES HERE
9578
9579  041404  062716  000006          130$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9580  041410  113776  041402  000000        MOVB    125$,@(SP)      ;MOVE ERROR # TO USERS CALL
9581  041416  000137  041636                JMP     215$
9582  041422                  135$:
9583  041422  004737  044346                JSR     PC,TIMOUT       ;WAIT FOR RECALIBRATE TO COMPLETE
9584  041426  004737  043536                JSR     PC,GET          ;GO GET RECAL STATUS
9585  041432  000411                        BR      150$            ;RETURN HERE IF NO ERROR
9586  041434  000401                        BR      145$            ;GET AROUND ERROR #
9587  041436  000000          140$:   .WORD                           ;ERROR # GOES HERE
9588
9589  041440  062716  000006          145$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9590  041444  113776  041436  000000        MOVB    140$,@(SP)      ;MOVE ERROR TO USERS CALL
9591  041452  000137  041636                JMP     215$
9592
9593  041456  004737  054472          150$:   JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE STATUS
9594  041462  000412                        BR      165$            ;RETURN HERE IF NO ERROR
9595  041464  000401                        BR      160$            ;GET AROUND ERROR #
9596  041466  000000          155$:   .WORD                           ;ERROR # GOES HERE
9597
9598  041470  005726          160$:   TST     (SP)+                   ;STRIP RETURN TO SUBROUTINE
```

```
9599  041472  062716  000006          ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9600  041476  113776  041466  000000  MOVB    155$,@(SP)      ;MOVE ERROR # TO USERS CALL
9601  041504  000137  041636          JMP     215$
9602
9603  041510                  165$:
9604
9605                          ;READ THE SECTOR IDENTIFIED BY RMDAO, INCLUDING HEADER AND DATA
9606
9607  041510  012737  000073  001400  MOV     #RH!GO,RMCS1O   ;LOAD READ HEADER AND DATA COMMAND
9608  041516  004737  044006          JSR     PC,PUT          ;PUT COMMAND
9609  041522  000411                  BR      180$            ;RETURN HERE IF NO ERROR
9610  041524  000401                  BR      175$            ;GET AROUND ERROR #
9611  041526  000000          170$:   .WORD                   ;ERROR # GOES HERE
9612
9613  041530  062716  000006  175$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9614  041534  113776  041526  000000  MOVB    170$,@(SP)      ;MOVE ERROR # TO USERS ERROR CALL
9615  041542  000137  041636          JMP     215$
9616
9617  041546  004737  044346  180$:   JSR     PC,TIMOUT       ;WAIT FOR READ OPERATION TO COMPLETE
9618  041552  004737  043536          JSR     PC,GET          ;GO GET STATUS FOR READ OPERATION
9619  041556  000411                  BR      195$            ;RETURN HERE IF NO ERROR
9620  041560  000401                  BR      190$            ;GET AROUND ERROR #
9621  041562  000000          185$:   .WORD                   ;ERROR # GOES HERE
9622
9623  041564  062716  000006  190$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9624  041570  113776  041562  000000  MOVB    185$,@(SP)      ;MOVE ERROR # TO CALL
9625  041576  000137  041636          JMP     215$
9626
9627  041602  004737  057036  195$:   JSR     PC,DTASTS       ;GO VERIFY RESULTS OF READ OPERATION
9628  041606  000412                  BR      210$            ;RETURN HERE IF NO ERROR
9629  041610  000401                  BR      205$            ;GET AROUND ERROR #
9630  041612  000000          200$:   .WORD                   ;ERROR # GOES HERE
9631
9632  041614  005726          205$:   TST     (SP)+           ;STRIP RETURN ADDRESS TO SUBROUTINE
9633  041616  062716  000006          ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
9634  041622  113776  041612  000000  MOVB    200$,@(SP)      ;MOVE ERROR # TO USERS CALL
9635  041630  000137  041636          JMP     215$
9636
9637  041634  000456          210$:   BR      240$
9638
9639  041636                  215$:
9640
9641                          ;AN ERROR HAS BEEN DETECTED IN TRYING TO READ THE BAD SECTOR FILE.
9642                          ;THE SECTOR WILL BE RETRIED IF POSSIBLE.
9643
9644  041636  032737  010000  001342          BIT     #MOL,RMDSI      ;IS MEDIUM ON LINE ??
9645  041644  001446                  BEQ     230$            ;YES !!
9646
9647  041646  005337  042626          DEC     500$            ;DECREMENT RETRY COUNT
9648  041652  100037                  BPL     225$            ;RETRY IF COUNT NOT NEG
9649
9650                          ;THE RETRY COUNT HAS EXPIRED - SEE IF THE ERROR IS MEDIA RELATED
9651  041654  013746  001344          MOV     RMER1I,-(SP)    ;GET ER1
9652  041660  042716  100720          BIC     #DCK!HCRC!HCE!FER!ECH,(SP)
9653  041664  032737  000010  001372  BIT     #DPE,RMER2I     ;WAS THER A DATA PARITY ERROR ??
9654  041672  001402                  BEQ     220$            ;NO !!
```

```
 9655  041674  042716  000010                 220$:  BIC     #PAR,(SP)          ;YES - CLEAR PAR
 9656  041700  005726                                 TST     (SP)+              ;ARE THERE ANY ERRORS NOT DUE TO MEDIA ?
 9657  041702  001027                                 BNE     230$               ;YES !!
 9658  041704  013746  001372                         MOV     RMER2I,-(SP)       ;GET ER2
 9659  041710  042726  100010                         BIC     #BSE!DPE,(SP)+     ;CLEAR MEDIA ERRORS
 9660  041714  001022                                 BNE     230$               ;BRANCH IF NONMEDIA ERRORS DETECTED
 9661
 9662                                         ;THE ERRORS DETECTED WHILE TRYING TO RECOVER THE BAD SECTOR FILE ARE
 9663                                         ;DUE TO THE MEDIA.  SEE IF THE BAD SECTOR FILE CAN BE RECOVERED FROM
 9664                                         ;ANOTHER AREA ON THE LAST TRACK
 9665
 9666  041716  062737  000002  001406                 ADD     #2,RMDAO           ;ADVANCE SECTOR ADDRESS
 9667  041724  122737  000012  001406                 CMPB    #10.,RMDAO         ;QUIT IF ALL MFG SECTORS HAVE BEEN
 9668  041732  001413                                 BEQ     230$               ;TRIED
 9669  041734  122737  000040  001406                 CMPB    #32.,RMDAO         ;QUIT IF ALL USER SECTORS HAVE
 9670  041742  001407                                 BEQ     230$               ;BEEN TRIED
 9671  041744  012737  000003  042626                 MOV     #3,500$            ;REINSTATE RETRY COUNT FOR THIS SECTOR
 9672  041752  162716  000006                 225$:   SUB     #6,(SP)            ;MOVE SP BACK TO NO ERROR
 9673  041756  000137  041104                         JMP     30$                ;RETRY THE READ OPERATION
 9674
 9675  041762                                 230$:
 9676
 9677                                         ;THE BAD SECTOR FILE CANNOT BE READ
 9678  041762  162716  000004                         SUB     #4,(SP)            ;MOVE SP TO ERROR RETURN
 9679  041766  000137  042622                         JMP     410$               ;GO TO MODULE EXIT
 9680
 9681  041772                                 240$:
 9682
 9683                                         ;THE SECTOR WAS RECOVERED WITHOUT ERROR -
 9684                                         ;       READ THE USER FILE IF THIS IS THE MFG FILE
 9685                                         ;       OR ELSE DONE
 9686
 9687  041772  122737  000011  001406                 CMPB    #9.,RMDAO          ;WAS THE USER FILE READ ??
 9688  042000  103451                                 BLO     260$               ;YES - READ IS COMPLETE
 9689
 9690                                         ;IF 144 IS IMPLEMENTED THEN READ THE USER FILE -
 9691                                         ;       ELSE DUMMY THE BAD SECTOR FILE
 9692  042002  032737  140000  105530                 BIT     #MSE!USE,MFGFIL    ;ARE THE BAD SECTOR FLAGS OFF ??
 9693  042010  001410                                 BEQ     250$               ;YES - 144 IS DISABLED
 9694
 9695  042012  012737  002012  001406                 MOV     #002012,RMDAO      ;READ THE USER FILE
 9696  042020  012737  000003  042626                 MOV     #3,500$            ;RELOAD THE RETRY COUNT FOR THIS SECTOR
 9697  042026  000137  041104                         JMP     30$                ;GO READ THE USER FILE
 9698  042032                                 250$:
 9699
 9700                                         ;DUMMY THE BAD SECTOR FILES
 9701  042032  010046                                 MOV     R0,-(SP)           ;;PUSH R0 ON STACK
 9702  042034  010146                                 MOV     R1,-(SP)           ;;PUSH R1 ON STACK
 9703  042036  012701  000374                         MOV     #252.,R1           ;R1 = NUMBER OF ENTRIES IN FILES
 9704  042042  012700  000014                         MOV     #14,R0             ;R0 = ADDRESS INDEX TO FILE  STORAGE
 9705  042046  012760  177777  105530  255$:          MOV     #-1,MFGFIL(R0)     ;ENTER ALL ONES IN MFG FILE
 9706  042054  012760  177777  106540                 MOV     #-1,USRFIL(R0)     ;ENTER ALL ONES IN USER FILE
 9707  042062  062700  000002                         ADD     #2,R0              ;ADVANCE ADDRESS
 9708  042066  005301                                 DEC     R1                 ;DECREMENT COUNT
 9709  042070  001366                                 BNE     255$               ;CONTINUE IF NOT DONE
 9710  042072  012701  000006                         MOV     #6.,R1             ;CLEAR ID, SERIAL NUMBER ETC
```

```
 9711  042076  005000                        CLR     R0
 9712  042100  005060  105530          256$: CLR     MFGFIL(R0)
 9713  042104  005060  106540                CLR     USRFIL(R0)
 9714  042110  062700  000002                ADD     #2,R0
 9715  042114  005301                        DEC     R1
 9716  042116  001370                        BNE     256$
 9717
 9718  042120  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 9719  042122  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 9720  042124                          260$:
 9721
 9722                          ;SET MEDIA ENABLE AND RESTORE THE USERS PUT BUFFER
 9723  042124  012737  177777  001474        MOV     #-1,MEDENB
 9724
 9725  042132  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 9726  042134  005000                        CLR     R0              ;R0 IS REGISTER INDEX
 9727  042136  016060  107550  001400  265$: MOV     BUFFER(R0),PUTBUF(R0)
 9728  042144  062700  000002                ADD     #2,R0           ;ADVANCE R0
 9729  042150  022700  000046                CMP     #46,R0          ;DONE ??
 9730  042154  103370                        BHIS    265$
 9731
 9732  042156  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 9733  042160                          270$:
 9734
 9735  042160                          300$:
 9736
 9737                          ;;*****************************************************************
 9738                          ;VERIFY THAT THE DESIRED SECTOR IS NOT IN THE MFG BAD SECTOR FILE
 9739                          ;AND NOT IN THE USERS BAD SECTOR FILE.  ASSIGN A NEW SECTOR IF THE
 9740                          ;SECTOR IS IN EITHER FILE.
 9741
 9742                          ;LOAD INITIAL VARIABLES AND COMPUTE THE NUMBER OF SECTORS
 9743  042160  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 9744  042162  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 9745  042164  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 9746  042166  013737  001434  001476        MOV     RMDC0,ASNDC     ;LOAD REQUESTED CYLINDER, TRACK,
 9747  042174  013737  001406  001500        MOV     RMDA0,ASNDA     ;AND SECTOR ADDRESS IN ASSIGNED STORAGE
 9748  042202  005002                        CLR     R2              ;R2 = NUMBER OF SECTORS
 9749  042204  013700  001402                MOV     RMWC0,R0        ;R0 = WORD COUNT
 9750  042210  005300                        DEC     R0              ;CONVERT FROM 2'S COMPLEMENT
 9751  042212  005100                        COM     R0
 9752  042214  012701  000400                MOV     #256.,R1        ;R1 = NUMBER OF WORDS PER SECTOR
 9753  042220  032737  000002  001400        BIT     #BIT1,RMCS10    ;IS THIS A HEADER AND DATA COMMAND ??
 9754  042226  001402                        BEQ     305$            ;NO !!
 9755  042230  012701  000402                MOV     #258.,R1        ;CHANGE WORDS PER SECTOR
 9756  042234  020100                  305$: CMP     R1,R0           ;IS THERE A FULL SECTOR ??
 9757  042236  101404                        BLOS    310$            ;YES !!
 9758  042240  005700                        TST     R0              ;IS R0 ZERO ??
 9759  042242  001405                        BEQ     315$            ;YES !!
 9760  042244  005202                        INC     R2              ;INCREMENT FOR PARTIAL SECTOR
 9761  042246  000403                        BR      315$
 9762  042250  160100                  310$: SUB     R1,R0           ;SUBTRACT ONE SECTOR FROM WORD COUNT
 9763  042252  005202                        INC     R2              ;INCREMENT SECTOR COUNT
 9764  042254  000767                        BR      305$
 9765  042256  010237  042626          315$: MOV     R2,500$         ;SAVE SECTOR COUNT
 9766
```

```
9767  042262                                316$:
9768
9769                                        ;LOAD PARAMETERS AND SEARCH THE MFG AND USER BAD SECTOR FILES FOR
9770                                        ;THE ASSIGNED SECTOR AND ADJACENT SECTORS IF THE SECTOR COUNT IS
9771                                        ;MORE THAN ONE.
9772
9773  042262  012737  105544  042636            MOV     #MFGFIL+14,540$ ;MOVE THE STARTING ADDRESS OF MFG
9774                                                                    ;FILE TO BASE ADDRESS STORAGE
9775  042270  013737  001476  042632  320$:     MOV     ASNDC,520$      ;LOAD COMPARING CYLINDER ADDRESS
9776  042276  013737  001500  042634            MOV     ASNDA,530$      ;LOAD COMPARING TRACK, SECTOR ADDRESS
9777  042304  013737  042626  042630            MOV     500$,510$       ;LOAD NUMBER OF SECTORS TO CONFIRM
9778
9779                                        ;SETUP FOR A BINARY SEARCH OF THE CURRENT FILE FOR THE COMPARING
9780                                        ;CYLINDER, TRACK AND SECTOR ADDRESS
9781  042312  013700  042636          325$:     MOV     540$,R0         ;LOAD THE BASE ADDRESS IN R0
9782  042316  012701  000376                    MOV     #<<127.*4>/2>,R1        ;R1 = LENGTH OF FILE
9783  042322  060100                            ADD     R1,R0           ;START BINARY DIVIDE AT CENTER
9784  042324  021037  042632          330$:     CMP     (R0),520$       ;DOES TABLE ENTRY = COMPARING CYLINDER ?
9785  042330  001405                            BEQ     345$            ;YES !!
9786  042332  101002                            BHI     340$            ;BRANCH IF ENTRY > COMPARING CYLINDER
9787  042334                          335$:
9788
9789                                        ;FILE ENTRY IS LESS THAN COMPARING CYLINDER (OR TRACK AND SECTOR),
9790                                        ;SO ADD DISPLACEMENT TO CURRENT POSITION
9791  042334  060100                            ADD     R1,R0
9792  042336  000410                            BR      350$
9793  042340                          340$:
9794
9795                                        ;FILE ENTRY IS GREATER THAN COMPARING CYLINDER SO SUBTRACT DISPLACEMENT
9796                                        ;FROM CURRENT POSITION
9797  042340  160100                            SUB     R1,R0
9798  042342  000406                            BR      350$
9799  042344                          345$:
9800
9801                                        ;FILE ENTRY EQUALS COMPARING CYLINDER.  SEE IF THE NEXT ENTRY EQUALS
9802                                        ;THE COMPARING TRACK, AND SECTOR.
9803  042344  026037  000002  042634            CMP     2(R0),530$      ;ARE THEY EQUAL ??
9804  042352  001413                            BEQ     360$            ;YES !!
9805  042354  101371                            BHI     340$            ;BRANCH IF HIGHER
9806  042356  000766                            BR      335$            ;BRANCH IF LOWER
9807  042360                          350$:
9808
9809                                        ;THE POINTER (R0) HAS BEEN ADJUSTED.  HALVE THE DISPLACEMENT AND
9810                                        ;CONTINUE THE SEARCH IF DISPLACEMENT NOT ZERO
9811  042360  005701                            TST     R1              ;IS DISPLACEMENT ZERO ??
9812  042362  001440                            BEQ     370$            ;YES !!
9813  042364  006201                            ASR     R1              ;HALVE THE DISLPACEMENT
9814  042366  042701  000003                    BIC     #3,R1
9815  042372  020037  042636                    CMP     R0,540$         ;IS THE NEW POINTER WITHIN BOUNDS ??
9816  042376  103432                            BLO     370$            ;NO !!
9817  042400  000751                            BR      330$            ;CONTINUE SEARCH
9818  042402                          360$:
9819
9820                                        ;THE COMPARING CYLINDER, TRACK AND SECTOR IS IN THE BAD SECTOR FILE.
9821                                        ;ADVANCE THE ASSIGNED SECTOR AND START THE SEARCH ALL OVER.
9822  042402  105237  001500                    INCB    ASNDA           ;INCREMENT SECTOR
```

```
 9823  042406  122737  000037  001500          CMPB    #31.,ASNDA      ;SECTOR OK ??
 9824  042414  103022                          BHIS    365$            ;YES !!
 9825  042416  105037  001500                  CLRB    ASNDA           ;CLEAR SECTOR AND ADVANCE TRACK
 9826  042422  105237  001501                  INCB    ASNDA+1
 9827  042426  122737  000004  001501          CMPB    #4,ASNDA+1      ;TRACK OK ??
 9828  042434  103012                          BHIS    365$            ;YES !!
 9829  042436  005037  001500                  CLR     ASNDA           ;CLEAR TRACK AND SECTOR
 9830  042442  005237  001476                  INC     ASNDC           ;INCREMENT CYLINDER
 9831  042446  022737  001466  001476          CMP     #822.,ASNDC     ;CYLINDER OK ??
 9832  042454  103002                          BHIS    365$            ;YES !!
 9833  042456  005037  001476                  CLR     ASNDC
 9834  042462  000677                  365$:   BR      316$            ;REPEAT SEARCH
 9835
 9836  042464                          370$:
 9837
 9838                                  ;THE COMPARING SECTOR IS NOT IN THE BAD SECTOR FILES.  DECREMENT THE
 9839                                  ;NUMBER OF SECTORS TO COMPARE AND SEARCH THE NEXT SECTOR IF THE NUMBER
 9840                                  ;IS NOT ZERO.
 9841
 9842  042464  005337  042630          DEC     510$            ;DECREMENT NUMBER OF SECTORS TO COMPARE
 9843  042470  001432                  BEQ     380$            ;DONE IF ZERO
 9844  042472  105237  042634          INCB    530$            ;INCREMENT THE COMPARING SECTOR
 9845  042476  122737  000037  042634  CMPB    #31.,530$       ;SECTOR OK ??
 9846  042504  103022                  BHIS    375$            ;YES !!
 9847  042506  105037  042634          CLRB    530$            ;CLEAR SECTOR
 9848  042512  105237  042635          INCB    530$+1          ;INCREMENT TRACK
 9849  042516  122737  000004  042635  CMPB    #4,530$+1       ;TRACK OK ??
 9850  042524  103012                  BHIS    375$            ;YES !!
 9851  042526  005037  042634          CLR     530$            ;CLEAR SECTOR TRACK
 9852  042532  005237  042632          INC     520$            ;INCREMENT CYLINDER
 9853  042536  022737  001466  042632  CMP     #822.,520$      ;CYLINDER OK ??
 9854  042544  103002                  BHIS    375$            ;YES !!
 9855  042546  005037  042632          CLR     520$            ;CLEAR CYLINDER
 9856  042552  000137  042312  375$:   JMP     325$            ;SEARCH NEXT SECTOR
 9857
 9858  042556                          380$:
 9859
 9860                                  ;THE ASSIGNED SECTOR (AND REQUIRED ADJACENT SECTORS) ARE NOT IN
 9861                                  ;THE BAD SECTOR FILE JUST SEARCHED.  SEARCH THE USER FILE IF IT
 9862                                  ;HAS NOT YET BEEN SEARCHED, ELSE ASSIGN THE SECTOR AND RETURN TO USER.
 9863
 9864  042556  022737  106554  042636          CMP     #USRFIL+14,540$ ;TEST BASE ADDRESS
 9865  042564  001405                          BEQ     400$            ;DONE IF BASE = USER
 9866  042566  012737  106554  042636          MOV     #USRFIL+14,540$ ;LOAD BASE ADDRESS
 9867  042574  000137  042270          390$:   JMP     320$            ;SEARCH USER FILE
 9868
 9869
 9870  042600                          400$:
 9871
 9872                                  ;ASSIGN THE SECTOR AND RETURN TO USER
 9873  042600  013737  001476  001434          MOV     ASNDC,RMDC0     ;LOAD CYLINDER
 9874  042606  013737  001500  001406          MOV     ASNDA,RMDA0     ;LOAD TRACK AND SECTOR
 9875  042614  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 9876  042616  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 9877  042620  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 9878  042622  000240                  410$:   NOP
```

```
 9879  042624  000207                        RTS     PC
 9880
 9881                                ;THE FOLLOWING ARE STORAGE LOCATIONS FOR THE MODULE
 9882
 9883  042626  000000                500$:   .WORD   ;RETRY COUNT/ NUMBER OF SECTORS REQUIRED
 9884  042630  000000                510$:   .WORD   ;NUMBER OF SECTORS TO COMPARE
 9885  042632  000000                520$:   .WORD   ;COMPARING CYLINDER
 9886  042634  000000                530$:   .WORD   ;COMPARING TRACK AND SECTOR
 9887  042636  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

042640                          GENBUF:
042640  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
042642  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
042644  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
042646  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
042650  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
042652  013700  001404                  MOV     RMBA0,R0        ;LOAD DATA BUFFER ADDRESS
042656  013701  001402                  MOV     RMWC0,R1        ;LOAD WORD COUNT
042662  013737  001434  043100          MOV     RMDC0,60$       ;LOAD STARTING CYLINDER ADDRESS
042670  013737  001406  043102          MOV     RMDA0,65$       ;LOAD STARTING TRACK,SECTOR ADDRESS
042676  032737  000002  001400  10$:    BIT     #BIT1,RMCS1     ;WRITE HEADER & DATA??
042704  001450                          BEQ     25$             ;NO!!
042706  013710  043100                  MOV     60$,(R0)        ;WRITE HEADER WORD #1
042712  042710  176000                  BIC     #^CCYLMSK,(R0)
042716  052710  140000                  BIS     #MSE!USE,(R0)   ;SET (DISABLE) BAD SECTOR FLAGS
042722  012702  000035                  MOV     #29.,R2         ;R2 = MAXIMUM SECTOR ADDRESS
042726  032737  010000  001432          BIT     #FMT16,RMOF0    ;16 BIT FORMAT??
042734  001404                          BEQ     15$             ;NO!!
042736  052710  010000                  BIS     #FMT16,(R0)     ;SET FORMAT BIT IN HEADER
042742  012702  000037                  MOV     #31.,R2         ;CHANGE MAXIMUM SECTOR ADDRESS
042746  005201                  15$:    INC     R1              ;INCREMENT WORD COUNT
042750  001444                          BEQ     50$             ;EXIT IF DONE
042752  062700  000002                  ADD     #2,R0           ;MOVE R0 TO HEADER WORD #2
042756  013720  043102                  MOV     65$,(R0)+       ;WRITE HEADER WORD #2
042762  005201                          INC     R1              ;INCREMENT WORD COUNT AND
042764  001436                          BEQ     50$             ;EXIT IF DONE
042766  012703  043102                  MOV     #65$,R3         ;ADVANCE SECTOR ADDRESS
042772  105213                          INCB    (R3)
042774  120213                          CMPB    R2,(R3)         ;SECTOR OVERFLOW ??
042776  103013                          BHIS    25$             ;NO !!
043000  105013                          CLRB    (R3)            ;YES - CLEAR SECTOR ADDRESS
043002  105263  000001                  INCB    1(R3)           ;ADVANCE TRACK ADDRESS
043006  122763  000004  000001          CMPB    #4,1(R3)        ;TRACK OVERFLOW ??
043014  103004                          BHIS    25$             ;NO !!
043016  105063  000001                  CLRB    1(R3)           ;YES - CLEAR TRACK ADDRESS
043022  105237  043100                  INCB    60$             ;ADVANCE CYLINDER ADDRESS
043026  012704  000400          25$:    MOV     #256.,R4                ;LOAD SECTOR DATA COUNT
043032  013702  001174          30$:    MOV     $TMP0,R2        ;LOAD PATTERN ADDRESS
043036  013703  001176                  MOV     $TMP1,R3        ;LOAD PATTERN COUNT
043042  012220                  40$:    MOV     (R2)+,(R0)+     ;WRITE DATA PATTERN
043044  005201                          INC     R1              ;INCREMENT WORD COUNT AND
043046  001405                          BEQ     50$             ;EXIT IF DONE
043050  005304                          DEC     R4              ;DECREMENT SECTOR COUNT
```

```
 9944  043052  001711                    BEQ     10$             ;START NEXT SECTOR IF 0
 9945  043054  005303                    DEC     R3              ;DECREMENT PATTERN COUNT
 9946  043056  001765                    BEQ     30$             ;RESTART PATTERN IF 0
 9947  043060  000770                    BR      40$             ;CONTINUE DATA PATTERN
 9948
 9949  043062                    50$:
 9950  043062  012604                    MOV     (SP)+,R4        ;;POP STACK INTO R4
 9951  043064  012603                    MOV     (SP)+,R3        ;;POP STACK INTO R3
 9952  043066  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
 9953  043070  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
 9954  043072  012600                    MOV     (SP)+,R0        ;;POP STACK INTO R0
 9955  043074  000240                    NOP
 9956  043076  000207                    RTS     PC
 9957
 9958  043100  000000            60$:    .WORD                   ;CYLINDER ADDRESS STORAGE
 9959  043102  000000            65$:    .WORD                   ;TRACK, SECTOR ADDRESS STORAGE
```

```
                                .SBTTL  COMPARE BUFFER SUBROUTINE

                                ;THIS SUBROUTINE COMPARES THE CONTENTS OF BUFONE AND BUFTWO,
                                ;ASSUMING THAT BUFONE IS THE BUFFER FROM WHICH DATA WAS WRITTEN
                                ;AND BUFTWO IS THE BUFFER TO WHICH DATA WAS READ.  ERRORS IN BUFFER
                                ; COMPARISON ARE REPORTED TO THE USER VIA THE USER'S ERROR CALL.

                                ;CALL:
                                ;(1)    JSR     PC,CMPBUF
                                ;(2)    .WORD   WRITE BUFFER ADDRESS
                                ;       .WORD   READ BUFFER ADDRESS
                                ;(3)    BR      ??              RETURN HERE IF NO ERROR
                                ;(4)    NOP                     RETURN HERE IF ERROR
                                ;(5)    ERROR                   ERROR DEFINED BY SUBROUTINE
                                ;(6)    ???

043104                          CMPBUF:
043104  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
043106  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
043110  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
043112  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK

043114  005037  043450                  CLR     150$            ;CLEAR CORRECTION FLAG
                                ;DETERMINE IF DATA SHOULD BE CORRECTED
043120  032737  100000  001344          BIT     #DCK,RMER1I     ;WAS THERE A DATA CHECK ??
043126  001465                          BEQ     80$             ;NO !!
043130  032737  000100  001344          BIT     #ECH,RMER1I     ;IS IT A HARD ERROR ??
043136  001061                          BNE     80$             ;YES !!
043140  033737  004000  001362          BIT     ECI,RMOFI       ;WAS ECC CORRECTION ALLOWED ??
043146  001055                          BNE     80$             ;NO !!
043150  032737  010000  001362          BIT     #FMT16,RMOFI    ;IS THIS 16 BIT FORMAT ??
043156  001451                          BEQ     80$             ;NO !!

                                ;CORRECT DATA USING ECC INFORMATION
043160  013700  001404                  MOV     RMBA0,R0        ;R0 = STARTING BUFFER ADDRESS
043164  013701  001374                  MOV     RMECI1,R1       ;R1 = ECC POSITION
043170  052737  100000  043450          BIS     #BIT15,150$     ;SET CORRECTION FLAG
                                ;MOVE R0 TO WORD BOUNDARY OF ERROR BURST

043176  022701  000020          10$:    CMP     #16.,R1         ;IS BIT POSITION > 1 WORD
043202  103005                          BHIS    20$             ;NO !!
043204  162701  000020                  SUB     #16.,R1         ;SUBTRACT 1 WORDS WORTH
043210  062700  000002                  ADD     #2,R0           ;ADVANCE BUFFER ADDRESS 1 WORD
043214  000770                          BR      10$
043216  012702  000001          20$:    MOV     #1,R2           ;R2 = BIT POINTER
043222  010203                          MOV     R2,R3           ;R3 = BIT NUMBER
                                ;MOVE R2 TO STARTING BIT OF ERROR BURST

043224  020301                  30$:    CMP     R3,R1           ;IS R3 SAME AS R1 ??
043226  001403                          BEQ     35$             ;YES !!
043230  006302                          ASL     R2              ;SHIFT BIT POINTER
043232  005203                          INC     R3              ;INCREMENT BIT NUMBER
043234  000773                          BR      30$
043236  012703  000013          35$:    MOV     #11.,R3         ;R3 = LENGTH OF ERROR BURST

                                ;CORRECT THE ERROR BURST
```

```
10016  043242  030237  001376          40$:    BIT     R2,RMEC2I       ;IS THIS BIT SET IN ECC PATTERN ??
10017  043246  001405                          BEQ     60$             ;NO - DO NOT CORRECT THIS BIT
10018  043250  030210                          BIT     R2,(R0)         ;IS THE BIT PRESENTLY SET ??
10019  043252  001402                          BEQ     50$             ;NO
10020  043254  040210                          BIC     R2,(R0)         ;RESET THE BIT
10021  043256  000401                          BR      60$
10022  043260  050210                  50$:    BIS     R2,(R0)         ;SET THE BIT
10023  043262  006302                  60$:    ASL     R2              ;SHIFT TO NEXT BIT
10024  043264  001004                          BNE     70$
10025  043266  012702  000001                  MOV     #1,R2           ;CONTINUE WITH FIRST BIT-OF NEXT WORD
10026  043272  062700  000002                  ADD     #2,R0
10027  043276  005303                  70$:    DEC     R3              ;END OF BURST ??
10028  043300  001360                          BNE     40$             ;NO !!
10029  043302  017600  000010          80$:    MOV     @10(SP),R0              ;R0 = WRITE BUFFER
10030  043306  062766  000002  000010          ADD     #2,10(SP)               ;MOVE SP TO READ ADDRESS
10031  043314  017601  000010                  MOV     @10(SP),R1              ;R1 = READ BUFFER
10032  043320  062766  000002  000010          ADD     #2,10(SP)               ;MOVE SP TO RETURN ADDRESS
10033  043326  013702  001332                  MOV     RMWCI,R2                ;R2 = NUMBER OF WORDS TRANSFER
10034  043332  163702  001402                  SUB     RMWCO,R2
10035  043336  022021                  90$:    CMP     (R0)+,(R1)+             ;COMPARE DATA WORDS
10036  043340  001003                          BNE     100$                    ;EXIT IF NOT EQUAL
10037  043342  005302                          DEC     R2                      ;DECREMENT WORD COUNT
10038  043344  001374                          BNE     90$                     ;CONTINUE IF NOT DONE
10039  043346  000433                          BR      110$                    ;DONE COMPARE - NO ERROR
10040  043350                          100$:
10041  043350  014037  001140                  MOV     -(R0),$GDDAT            ;STORE GOOD DATA FOR TYPEOUT
10042  043354  014137  001142                  MOV     -(R1),$BDDAT            ;STORE BAD DATA FOR TYPEOUT
10043  043360  010037  001134                  MOV     R0,$GDADR               ;STORE ADDRESS OF GOOD DATA
10044  043364  010137  001136                  MOV     R1,$BDADR               ;STORE ADDRESS OF BAD DATA
10045  043370  010237  001174                  MOV     R2,$TMP0                ;STORE WORD COUNT OF ERROR
10046  043374  062766  000004  000010          ADD     #4,10(SP)               ;MOVE SP TO USER'S ERROR CALL
10047  043402  112776  000336  000010          MOVB    #336,@10(SP)            ;WRITE ERROR NUMBER IN CALL
10048
10049                          ;CHANGE ERROR NUMBER IF ECC CORRECTION FAILED
10050  043410  032737  100000  043450          BIT     #BIT15,150$     ;WAS ECC CORRECTION USED ??
10051  043416  001402                          BEQ     105$            ;NO !!
10052  043420  112776  000163  000010          MOVB    #163,@10(SP)    ;ECC CORRECTION FAILED
10053  043426  162766  000002  000010  105$:   SUB     #2,10(SP)               ;MOVE SP TO RETURN IF ERROR
10054  043434  000240                          NOP
10055  043436                          110$:
10056  043436  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
10057  043440  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
10058  043442  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10059  043444  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10060  043446  000207                          RTS     PC                      ;RETURN TO USER
10061
10062  043450  000000          150$:   .WORD                           ;ECC CORRECTION FLAG
```

```
10063                                   .SBTTL  GET STATUS SUBROUTINE
10064                                   ;
10065                                   ;THIS SUBROUTINE SETS UP THE "GET INDEX TABLE" AND THE "GET
10066                                   ;BUFFER" FOR READING ALL SUBSYSTEM REGISTERS VIA THE GET SUBROUTINE
10067                                   ;AND THEN RETURNS TO THE USER.
10068                                   ;
10069                                   ;CALL:  JSR     PC,GETSTS
10070                                   ;       ???                     RETURN HERE
10071
10072   043452                          GETSTS:
10073   043452  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10074   043454  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10075   043456  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10076   043460  012700  001506                  MOV     #GETINX,R0      ;R0= ADDRESS OF INDEX TABLE
10077   043464  012701  001400                  MOV     #RMEC2I+2,R1    ;R1 = ADDRESS OF GET BUFFER
10078   043470  012702  000046                  MOV     #RMEC2,R2       ;R2 = REGISTER INDE
10079   043474  110220                  2$:     MOVB    R2,(R0)+        ;WRITE REGISTER INDEX IN TABLE
10080   043476  005041                          CLR     -(R1)           ;CLEAR CORRESPONDING LOCATION
10081   043500  162702  000002          3$:     SUB     #2,R2           ;DECREMENT TO NEXT INDEX
10082   043504  100405                          BMI     4$              ;BRANCH OUT IF DONE
10083   043506  022702  000022                  CMP     #RMDB,R2        ;DONT WRITE RMDB INDEX
10084   043512  001370                          BNE     2$
10085   043514  005041                          CLR     -(R1)
10086   043516  000770                          BR      3$
10087   043520  112720  000200          4$:     MOVB    #200,(R0)+      ;WRITE TERMINATOR
10088   043524  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
10089   043526  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10090   043530  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10091   043532  000240                          NOP
10092   043534  000207                          RTS     PC
10093
```

```
10094                                   .SBTTL  GET SUBROUTINE
10095
10096                                   ;THIS SUBROUTINE READS THE REGISTERS WHICH ARE LISTED IN THE
10097                                   ;"GET INDEX TABLE" AND STORES THEIR VALUES IN THE CORRESPONDING
10098                                   ;LOCATION IN THE "GET REGISTER BUFFER".  FOR EXAMPLE, AN
10099                                   ;ENTRY OF 04 IN THE TABLE WILL CAUSE THE SUBROUTINE TO
10100                                   ;READ "RMBA" AND STORE ITS CONTENTS AT THE LOCATION IN
10101                                   ;THE BUFFER ASSIGNED TO THAT REGISTER.  THE NUMBER OF
10102                                   ;REGISTERS TO BE READ IS VARIABLE FROM 1 TO 22; THE INDEX
10103                                   ;TABLE MUST BE TERMINATED WITH A CONTROL BYTE (200)
10104                                   ;WHICH SHOULD FOLLOW THE LAST ENTRY.
10105                                   ;
10106                                   ;SUBROUTINE CALL:
10107                                   ;(1)    "GET INDEX TABLE" HAS BEEN LOADED WITH REGISTER INDEX
10108                                   ;       VALUES AND TERMINATED WITH A CONTROL BYTE
10109                                   ;(2)    "GET INPUT BUFFER" IS AVAILABLE FOR USE.  (NOTE THAT
10110                                   ;       UNUSED LOCATIONS, I.E.,ENTRIES IN BUFFER CORRESPONDING
10111                                   ;       TO REGISTERS NOT READ, ARE NOT CHANGED.)
10112                                   ;(3)    JSR     PC,GET
10113                                   ;       BR      ???             RETURN HERE IF NO ERROR FOUND
10114                                   ;       NOP                     RETURN HERE IF ANY ERROR FOUND
10115                                   ;       ERROR                   SUB DEFINES ERROR NUMBER
10116                                   ;       ???
10117
10118  043536  000240                   GET:    NOP
10119  043540  062716  000004                   ADD     #4,(SP)         ;CLEAR ERROR NUMBER IN USER'S
10120  043544  105076  000000                   CLRB    @(SP)           ;ERROR CALL
10121  043550  162716  000004                   SUB     #4,(SP)
10122  043554  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10123  043556  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10124  043560  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10125  043562  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
10126  043564  010446                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
10127  043566  013746  000004                 MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10128  043572  013746  000006                 MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10129  043576  013700  001276                 MOV     $BASE,R0
10130  043602  012702  001330                 MOV     #GETBUF,R2
10131  043606  012704  001506                 MOV     #GETINX,R4
10132  043612  012737  043720  000004         MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
10133  043620  012737  000300  000006         MOV     #PR6,ERRVEC+2
10134  043626  016037  000010  001174  1$:    MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
10135  043634  016037  000000  001176         MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
10136  043642  032737  004000  001176         BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
10137  043650  001007                         BNE     3$              ;YES!!
10138  043652  062766  000004  000016         ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN USER'S
10139  043660  112776  000112  000016         MOVB    #112,@16(SP)    ;ERROR CALL
10140  043666  000423                         BR      7$
10141  043670  105714                 3$:     TSTB    (R4)            ;DONE??
10142  043672  100433                         BMI     9$              ;YES!!
10143  043674  111401                         MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
10144  043676  042701  177700                 BIC     #^CIDXMSK,R1    ;CLEAR ANY SIGN EXTENSION
10145  043702  060001                         ADD     R0,R1
10146  043704  112403                         MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS FOR REGISTER
10147  043706  042703  177700                 BIC     #^CIDXMSK,R3    ;CLEAR ANY SIGN EXTENSION
10148  043712  060203                         ADD     R2,R3
10149  043714  011113                         MOV     (R1),(R3)       ;READ REGISTER
```

```
10150  043716  000764                          BR      3$
10151
10152  043720  022626                  5$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
10153  043722  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
10154  043730  112776  000007  000016          MOVB    #7,@16(SP)      ;USER'S ERROR CALL
10155  043736  162766  000002  000016  7$:     SUB     #2,16(SP)
10156  043744  105714                  8$:     TSTB    (R4)            ;DONE CLEARING??
10157  043746  100405                          BMI     9$              ;YES!!
10158  043750  005003                          CLR     R3              ;CLEAR REMAINING STORAGE
10159  043752  112403                          MOVB    (R4)+,R3        ;LOCATIONS
10160  043754  060203                          ADD     R2,R3
10161  043756  005013                          CLR     (R3)
10162  043760  000771                          BR      8$
10163  043762                          9$:
10164  043762  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10165  043766  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10166  043772  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
10167  043774  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
10168  043776  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
10169  044000  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10170  044002  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10171  044004  000207                          RTS     PC              ;RETURN
10172
```

```
10173                                   .SBTTL  PUT SUBROUTINE
10174
10175                                   ;THIS SUBROUTINE WRITES THE REGISTERS WHICH ARE LISTED IN THE
10176                                   ;"PUT INDEX TABLE" WITH  THE CONTENTS OF THE CORRESPONDING
10177                                   ;LOCATION IN THE "PUT REGISTER BUFFER".  THE NUMBER OF
10178                                   ;REGISTERS WRITTEN IS VARIABLE; THE INDEX TABLE MUST
10179                                   ;BE TERMINATED WITH A CONTROL BYTE (200) WHICH SHOULD
10180                                   ;FOLLOW THE LAST ENTRY.
10181                                   ;
10182                                   ;SUBROUTINE CALL:
10183                                   ;
10184                                   ;(1)    "PUT INDEX TABLE" HAS BEEN LOADED WITH INDEX VALUES
10185                                   ;       OF REGISTERS TO BE WRITTEN.
10186                                   ;(2)    "PUT REGISTER BUFFER" CONTAINS CONTENTS OF EACH
10187                                   ;       REGISTER TO BE WRITTEN.
10188                                   ;(3)    JSR     PC,PUT
10189                                   ;       BR      ???             RETURN HERE IF NO ERROR FOUND
10190                                   ;       NOP                     RETURN HERE IF ANY ERROR FOUND
10191                                   ;       ERROR                   SUB DEFINES ERROR NUMBER
10192                                   ;       ???
10193
10194  044006  000240                   PUT:    NOP
10195  044010  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10196  044012  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10197  044014  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10198  044016  010346                           MOV     R3,-(SP)        ;;PUSH R3 ON STACK
10199  044020  010446                           MOV     R4,-(SP)        ;;PUSH R4 ON STACK
10200  044022  013746  000004                   MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10201  044026  013746  000006                   MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10202  044032  013700  001276                   MOV     $BASE,R0
10203  044036  012702  001400                   MOV     #PUTBUF,R2
10204  044042  012704  001535                   MOV     #PUTINX,R4
10205  044046  012737  044154  000004           MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
10206  044054  012737  000300  000006           MOV     #PR6,ERRVEC+2
10207  044062  016037  000010  001174  1$:      MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
10208  044070  016037  000000  001176           MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
10209  044076  032737  004000  001176           BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
10210  044104  001007                           BNE     3$              ;YES!!
10211  044106  062766  000004  000016           ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
10212  044114  112776  000112  000016           MOVB    #112,@16(SP)    ;USER'S ERROR CALL
10213  044122  000423                           BR      7$
10214  044124  105714                  3$:      TSTB    (R4)            ;DONE??
10215  044126  100424                           BMI     9$              ;YES!!
10216  044130  111401                           MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
10217  044132  042701  177700                   BIC     #^CIDXMSK,R1    ;CLEAR ANY SIGN EXTENSION
10218  044136  060001                           ADD     R0,R1
10219  044140  112403                           MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS
10220  044142  042703  177700                   BIC     #^CIDXMSK,R3    ;CLEAR ANY SIGN EXTENSION
10221  044146  060203                           ADD     R2,R3
10222  044150  011311                           MOV     (R3),(R1)       ;WRITE REGISTER
10223  044152  000764                           BR      3$
10224
10225  044154  022626                  5$:      CMP     (SP)+,(SP)+     ;ADJUST STACK
10226  044156  062766  000004  000016           ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
10227  044164  112776  000007  000016           MOVB    #7,@16(SP)      ;USER'S ERROR CALL
10228  044172  162766  000002  000016  7$:      SUB     #2,16(SP)
```

```
10229
10230  044200                          9$:
10231  044200  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10232  044204  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10233  044210  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
10234  044212  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
10235  044214  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
10236  044216  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10237  044220  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
10238  044222  000207                          RTS     PC              ;RETURN
10239
```

```
10240                                           .SBTTL  SIZE CLOCK SUBROUTINE
10241
10242  044224                                   SIZCLK:
10243  044224  013746  000004                          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10244  044230  013746  000006                          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10245  044234  012737  044272  000004                  MOV     #1$,ERRVEC      ;SET UP FOR BUS TIMEOUT
10246  044242  012737  000300  000006                  MOV     #PR6,ERRVEC+2
10247  044250  012737  177546  001502                  MOV     #177546,CLKADR  ;LOAD ADDRESSES FOR KW11-L
10248  044256  012737  000100  001504                  MOV     #100,CLKVCT
10249  044264  005777  135212                         TST     @CLKADR         ;TEST FOR KW11-L PRESENT
10250  044270  000421                                 BR      3$              ;YES - KW11-L IS PRESENT
10251  044272  022626                           1$:    CMP     (SP)+,(SP)+     ;RESTORE SP
10252  044274  012737  044324  000004                  MOV     #2$,ERRVEC      ;SET UP FOR BUS TIMEOUT
10253  044302  012737  172540  001502                  MOV     #172540,CLKADR  ;LOAD ADDRESSES FOR KW11-P CLOCK
10254  044310  012737  000104  001504                  MOV     #104,CLKVCT
10255  044316  005777  135160                         TST     @CLKADR         ;TEST FOR KW11-P PRESENT
10256  044322  000404                                 BR      3$              ;YES - KW11-P IS PRESENT
10257  044324  022626                           2$:    CMP     (SP)+,(SP)+     ;RESTORE SP
10258  044326  062766  000002  000004                  ADD     #2,4(SP)        ;MOVE RETURN TO ERROR
10259  044334                                   3$:
10260  044334  012637  000006                          MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10261  044340  012637  000004                          MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10262  044344  000207                                 RTS     PC              ;RETURN TO USER
```

```
10263                                   .SBTTL  TIMEOUT SUBROUTINE
10264                                   ;
10265                                   ;THIS SUBROUTINE WAITS FOR GO TO RESET OR FOR A TIMEOUT GREATER THAN
10266                                   ;500 MS, WHICH EVER OCCURRS FIRST, AND THEN RETURNS.
10267                                   ;
10268                                   ;CALL:  JSR     PC,TIMOUT
10269                                   ;       ???                     RETURN HERE
10270                                   ;
10271   044346                          TIMOUT:
10272   044346  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10273   044350  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10274   044352  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10275   044354  013746  000004                 MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10276   044360  013746  000006                 MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10277   044364  012737  044466  000004         MOV     #4$,ERRVEC      ;SETUP FOR BUS TIMEOUT - 04 TRAP
10278   044372  012737  000300  000006         MOV     #PR6,ERRVEC+2
10279   044400  013700  001276                 MOV     $BASE,R0        ;R0=RM03 ADDRESS
10280   044404  013701  001502                 MOV     CLKADR,R1       ;R1=CLOCK ADDRESS
10281   044410  012702  000037                 MOV     #31.,R2         ;R2=NUMBER OF CLOCK CYCLES
10282   044414  020127  172540          1$:    CMP     R1,#172540      ;KW11-P CLOCK??
10283   044420  001003                         BNE     2$              ;NO!!
10284   044422  012761  000001  000002         MOV     #1,2(R1)        ;SET COUNTER
10285   044430  012711  000005          2$:    MOV     #BIT2!BIT0,(R1) ;START COUNTER
10286
10287   044434  016046  000000          3$:    MOV     RMCS1(R0),-(SP) ;GET STATUS
10288   044440  042716  177576                 BIC     #^C<RDY!GO>,(SP)
10289   044444  022726  000200                 CMP     #RDY,(SP)+      ;RDY=1,GO=0??
10290   044450  001420                         BEQ     5$              ;YES!!
10291   044452  032711  000200                 BIT     #BIT7,(R1)      ;TIMER DONE??
10292   044456  001766                         BEQ     3$              ;NO!!
10293   044460  005302                         DEC     R2              ;DEC NUMBER OF CYCLES
10294   044462  001354                         BNE     1$              ;CONTINUE IF NOT DONE
10295   044464  000412                         BR      5$
10296   044466  022626          4$:    CMP     (SP)+,(SP)+     ;ADJUST STACK
10297   044470  062766  000004  000012         ADD     #4,12(SP)       ;MOVE SP TO USER'S CALL
10298   044476  112776  000007  000012         MOVB    #7,@12(SP)      ;WRITE ERROR NUMBER
10299   044504  162766  000002  000012         SUB     #2,12(SP)
10300
10301   044512                          5$:
10302   044512  012637  000006                 MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10303   044516  012637  000004                 MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10304   044522  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
10305   044524  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
10306   044526  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
10307   044530  000207                         RTS     PC              ;RETURN TO USER
10308
```

```
10309                                   .SBTTL  PRIMARY ERROR CHECK SUBROUTINE
10310                                   ;
10311                                   ;THE PURPOSE OF THIS SUBROUTINE IS TO VERIFY THAT STAUTS IS VALID AND
10312                                   ;THAT FURTHER ERROR AND STATUS CHECKING SHOULD BE PERFORMED.  THE
10313                                   ;FOLLOWING CHECKS ARE MADE:
10314                                   ;
10315                                   ;       .CURRECT UNIT IS SELECTED, I.E., THE UNIT SELECT BITS OF RMCS2
10316                                   ;(BITS 0-2) EQUAL THE UNIT BEING TESTED;
10317                                   ;
10318                                   ;       .SELECTED UNIT IS AVAILABLE, I.E., DVA (BIT 11 OF RMCS1) IS SET
10319                                   ;AND NED (BIT 12 OF RMCS2) IS RESET;
10320                                   ;
10321                                   ;       .LAST COMMAND WAS COMPLETED, I.E., THE MASSBUS CONTROLLER IS
10322                                   ;READY (BIT 7 OF RMCS1) AND THE GO BIT IS RESET (BIT 0 OF RMCS1) OR THE
10323                                   ;DRIVE READY BIT (BIT 7 OF RMDS) IS SET.
10324                                   ;       .NO PARITY ERROR OCCURRED WHEN READING REMOTE REGISTERS,
10325                                   ;I.E., MCPE = 0.
10326                                   ;       .NO PARITY ERROR OCCURRED WHEN WRITING REMOTE REGISTERS,
10327                                   ;I.E., PAR = 0, OR, PAR = DPE = 1
10328                                   ;
10329                                   ;
10330                                   ;THE SUBROUTINE ASSUMES THAT:
10331                                   ;
10332                                   ;       .STATUS HAS BEEN STORED IN THE REGISTER INPUT BUFFER,
10333                                   ;IN PARTICULAR, RMCS1, RMCS2 AND RMDS HAVE BEEN STORED IN THEIR
10334                                   ;CORRESPONDING LOCATIONS OF THE "GET" BUFFER.
10335                                   ;
10336                                   ;       .($UNIT) CONTAINS THE DRIVE NUMBER
10337                                   ;
10338                                   ;THE SUBROUTINE IS CALLED AS FOLLOWS:
10339                                   ;
10340                                   ;(1)    JSR     PC,PRIERR
10341                                   ;       BR      ???             RETURN HERE IF NO ERROR
10342                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10343                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10344                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10345                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10346
10347   044532                         PRIERR:
10348
10349                                   ;CLEAR USER'S ERROR CALL
10350   044532  062716  000004                ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
10351   044536  105076  000000                CLRB    @(SP)           ;CLEAR ERROR NUMBER
10352   044542  162716  000004                SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10353
10354
10355                                   ;REPORT AN ERROR IF THE WRONG UNIT IS SELECTED
10356   044546  013737  001340  001142        MOV     RMCS2I,$BDDAT   ;CORRECT UNIT SELECTED??
10357   044554  042737  177770  001142        BIC     #^CUNTMSK,$BDDAT
10358   044562  013737  001234  001140        MOV     $UNIT,$GDDAT    ;GOOD DATA FOR TYPEOUT
10359   044570  042737  177770  001140        BIC     #^CUNTMSK,$GDDAT
10360   044576  123737  001140  001142        CMPB    $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
10361                                                                   ;DRIVE NUMBERS
10362   044604  001415                        BEQ     1$              ;YES!!
10363   044606  062716  000004                ADD     #4,(SP)
10364   044612  112776  000001  000000        MOVB    #1,@(SP)        ;ERROR 1
```

```
10365  044620  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10366  044624  004736                          JSR     PC,@(SP)+        ;REPORT WRONG UNIT SELECTED
10367  044626  162716  000010                  SUB     #10,(SP)         ;RESTORE (SP)
10368  044632  000240                          NOP
10369  044634  000137  045354                  JMP     10$              ;SKIP OTHER CHECKS
10370  044640                          1$:
10371
10372                                          ;REPORT AN ERROR IF THE DEVICE IS NOT AVAILABLE OR IF
10373                                          ;THE DEVICE IS NONEXISTANT
10374  044640  032737  004000  001330          BIT     #DVA,RMCS1I      ;DEVICE AVAILABLE??
10375  044646  001045                          BNE     5$               ;YES!!
10376  044650  013737  001330  001140          MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
10377  044656  052737  004000  001140          BIS     #DVA,$GDDAT
10378  044664  013737  001330  001142          MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
10379  044672  062716  000004                  ADD     #4,(SP)
10380  044676  112776  000002  000000          MOVB    #2,@(SP)         ;ERROR #2
10381  044704  032737  010000  001340          BIT     #NED,RMCS2I      ;WAS NED SET??
10382  044712  001414                          BEQ     2$               ;NO!!
10383  044714  013737  001340  001140          MOV     RMCS2I,$GDDAT    ;EXPECTED STATUS
10384  044722  013737  001340  001142          MOV     RMCS2I,$BDDAT    ;RECEIVED STATUS
10385  044730  042737  010000  001140          BIC     #NED,$GDDAT
10386  044736  112776  000003  000000          MOVB    #3,@(SP)         ;YES - CHANGE ERROR NUMBER
10387  044744  162716  000002          2$:     SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10388  044750  004736                          JSR     PC,@(SP)+        ;REPORT DEVICE NOT AVAILABLE
10389  044752  162716  000010                  SUB     #10,(SP)         ;RESTORE (SP)
10390  044756  000240                          NOP
10391  044760  000575                          BR      10$              ;SKIP OTHER CHECKS
10392  044762                          5$:
10393
10394                                          ;REPORT AN ERROR IF MASSBUS CONTROLLER IS NOT READY
10395  044762  032737  000200  001330          BIT     #RDY,RMCS1I      ;CONTROLLER READY??
10396  044770  001030                          BNE     7$               ;YES!!
10397  044772  013737  001330  001140          MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
10398  045000  052737  000200  001140          BIS     #RDY,$GDDAT
10399  045006  042737  160001  001140          BIC     #SC!TRE!MCPE!GO,$GDDAT
10400  045014  013737  001330  001142          MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
10401  045022  062716  000004                  ADD     #4,(SP)
10402  045026  112776  000004  000000          MOVB    #4,@(SP)         ;ERROR #4
10403  045034  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10404  045040  004736                          JSR     PC,@(SP)+        ;REPORT CONTROLLER NOT READY
10405  045042  162716  000010                  SUB     #10,(SP)         ;RESTORE (SP)
10406  045046  000240                          NOP
10407  045050  000541                          BR      10$              ;SKIP OTHER CHECKS
10408  045052                          7$:
10409
10410                                          ;REPORT AN ERROR IF GO IS NOT ZERO AND DRY IS NOT ONE
10411  045052  032737  000001  001330          BIT     #GO,RMCS1I       ;GO RESET??
10412  045060  001431                          BEQ     8$               ;YES!!
10413  045062  032737  000200  001342          BIT     #DRY,RMDSI       ;DRIVE READY??
10414  045070  001025                          BNE     8$               ;YES!!
10415  045072  013737  001330  001140          MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
10416  045100  042737  160001  001140          BIC     #SC!TRE!MCPE!GO,$GDDAT
10417  045106  013737  001330  001142          MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
10418  045114  062716  000004                  ADD     #4,(SP)
10419  045120  112776  000005  000000          MOVB    #5,@(SP)         ;ERROR #5
10420  045126  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
```

```
10421  045132  004736                        JSR     PC,@(SP)+       ;REPORT DRIVE NOT READY
10422  045134  162716  000010                SUB     #10,(SP)        ;RESTORE (SP)
10423  045140  000240                        NOP
10424  045142  000504                        BR      10$
10425  045144                          8$:
10426
10427                                  ;REPORT AN ERROR IF THE RH CONTROLLER DETECTED BAD
10428                                  ;PARITY ON THE MASSBUS CONTROL BUS
10429  045144  032737  020000  001330          BIT     #MCPE,RMCS1I    ;PARITY ERROR ??
10430  045152  001425                        BEQ     9$              ;NO!!
10431  045154  013737  001330  001140        MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
10432  045162  042737  160001  001140        BIC     #SC!TRE!MCPE!GO,$GDDAT
10433  045170  013737  001330  001142        MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
10434  045176  062716  000004                ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
10435  045202  112776  000013  000000        MOVB    #13,@(SP)       ;ERROR #47
10436  045210  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10437  045214  004736                        JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10438  045216  162716  000010                SUB     #10,(SP)        ;RESTORE STACK
10439  045222  000240                        NOP
10440  045224  000453                        BR      10$
10441  045226                          9$:
10442
10443                                  ;REPORT AN ERROR IF THE RM03 DETECTED A CONTROL BUS PARITY ERROR
10444  045226  032737  000010  001344          BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
10445  045234  001451                        BEQ     11$             ;NO!!
10446  045236  032737  000010  001372        BIT     #DPE,RMER2I     ;WAS IT THE CONTROL BUS??
10447  045244  001045                        BNE     11$             ;NOT SURE!!
10448  045246  032737  000010  001414        BIT     #PAR,RMER1O     ;DID TEST SET PAR ??
10449  045254  001413                        BEQ     93$             ;NO!!
10450  045256  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10451  045260  012700  001535                MOV     #PUTINX,R0      ;R0 POINTS TO INDEX TABLE
10452  045264  122710  000014          91$:  CMPB    #RMER1,(R0)     ;SEARCH TABLE FOR RMER1
10453  045270  001002                        BNE     92$
10454  045272  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
10455  045274  000431                        BR      11$             ;PAR WAS SET BY TEST
10456  045276  105720                  92$:  TSTB    (R0)+           ;END OF TABLE??
10457  045300  100371                        BPL     91$             ;NO!!
10458  045302  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
10459  045304  013737  001344  001140  93$:  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10460  045312  042737  000010  001140        BIC     #PAR,$GDDAT
10461  045320  013737  001344  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10462  045326  062716  000004                ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10463  045332  112776  000050  000000        MOVB    #50,@(SP)       ;WRITE THE ERROR NUMBER
10464  045340  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10465  045344  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR
10466  045346  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10467  045352  000240                        NOP
10468  045354  062716  000010          10$:  ADD     #10,(SP)                ;RETURN TO ERROR
10469  045360  000240                  11$:  NOP                     ;RETURN TO NO ERROR
10470  045362  000207                        RTS     PC
10471
```

```
10472                                   .SBTTL  SECONDARY ERROR CHECK SUBROUTINE
10473                                   ;
10474                                   ;THE ERROR CHECK SUBROUTINE PROVIDES DETECTION OF SECONDARY ERRORS
10475                                   ;SUCH AS UNEXPECTED ERRORS AND UNEXPECTED REGISTER
10476                                   ;CONTENTS.  THESE ERRORS ARE DEEMED
10477                                   ;SECONDARY IN THAT THEY ARE NOT NECESSARILY ASSOCIATED WITH THE OPERATION
10478                                   ;BEING PERFORMED.  WHEN THE SUBROUTINE IDENTIFIES SUCH AN ERROR, IT MOVES
10479                                   ;THE ERROR NUMBER TO THE ERROR CALL IN THE TEST ROUTINE AND THEN RETURNS
10480                                   ;TO THE TEST ROUTINE WHICH MAKES THE ERROR CALL.  AFTER THE TEST ROUTINE
10481                                   ;MAKES THE ERROR CALL, IT RETURNS TO THE SUBROUTINE WHICH THEN LOOKS FOR
10482                                   ;OTHER ERRORS.  WHEN ALL ERRORS HAVE BEEN REPORTED, THE SUBROUTINE
10483                                   ;RETURNS TO THE ADDRESS FOLLOWING THE SUBROUTINE CALL.
10484                                   ;
10485                                   ;CALL:  JSR     PC,SECERR
10486                                   ;       BR      ???             RETURN HERE IF NO ERROR
10487                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10488                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10489                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10490                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10491                                   ;
10492                                   ;NOTE:  THE SUBROUTINE ASSUMES THAT REGISTERS HAVE BEEN STORED AT THE
10493                                   ;INPUT REGISTER BUFFER.
10494
10495  045364                           SECERR:
10496
10497                                   ;;*****************************************************************
10498                                   ;STORE FUNCTION CODE AND CLEAR USER'S ERROR NUMBER
10499  045364  013737  001400  051224           MOV     RMCS10,5I5$     ;STORE FUNCTION CODE
10500  045372  042737  177701  051224           BIC     #^C<F0!F1!F2!F3!F4>,5I5$
10501  045400  062716  000004                   ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
10502  045404  105076  000000                   CLRB    @(SP)           ;CLEAR ERROR NUMBER
10503  045410  162716  000004                   SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10504
10505                                   ;;*****************************************************************
10506                                   ;CHECK SECONDARY ERRORS COMMON TO ALL COMMANDS
10507
10508                                   ;REPORT ERROR IF DRIVE IS NOT READY, I.E., IF DRY = 0
10509  045414  032737  000200  001342           BIT     #DRY,RMDSI      ;DRIVE READY??
10510  045422  001024                           BNE     5$              ;YES!!
10511  045424  013737  001342  001142           MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
10512  045432  042737  177577  001142           BIC     #^CDRY,$BDDAT
10513  045440  012737  000200  001140           MOV     #DRY,$GDDAT     ;GOOD DATA FOR TYPEOUT
10514  045446  062716  000004                   ADD     #4,(SP)
10515  045452  112776  000010  000000           MOVB    #10,@(SP)       ;ERROR NUMBER
10516  045460  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10517  045464  004736                           JSR     PC,@(SP)+       ;REPORT NOT READY
10518  045466  162716  000010                   SUB     #10,(SP)                ;RESTORE (SP) TO ERROR N
10519  045472  000240                           NOP
10520
10521                                   ;REPORT ERROR IF GO BIT IS NOT RESET
10522  045474  032737  000001  001330  5$:      BIT     #GO,RMCS1I      ;GO BIT RESET??
10523  045502  001423                           BEQ     10$             ;YES!!
10524  045504  013737  001330  001142           MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
10525  045512  042737  177776  001142           BIC     #^CGO,$BDDAT
10526  045520  005037  001140                   CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
10527  045524  062716  000004                   ADD     #4,(SP)
```

```
10528  045530  112776  000011  000000          MOVB    #11,@(SP)       ;ERROR NUMBER
10529  045536  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10530  045542  004736                          JSR     PC,@(SP)+       ;REPORT DEVICE NOT AVAILABLE
10531  045546  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10532  045550  000240                          NOP
10533
10534                          ;REPORT ERROR IF FUNCTION CODE READ FROM DEVICE IS NOT CORRECT
10535  045552  013737  001330  001142  10$:     MOV     RMCS1I,$BDDAT   ;IS FUNCTION CODE CORRECT??
10536  045560  042737  177701  001142          BIC     #^C76,$BDDAT
10537  045566  013737  051224  001140          MOV     515$,$GDDAT     ;EXPECTED FUNCTION CODE
10538  045574  023737  001142  001140          CMP     $BDDAT,$GDDAT
10539  045602  001413                          BEQ     15$             ;YES!!
10540  045604  062716  000004                  ADD     #4,(SP)
10541  045610  112776  000012  000000          MOVB    #12,@(SP)       ;ERROR NUMBER
10542  045616  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10543  045622  004736                          JSR     PC,@(SP)+       ;REPORT WRONG FUNCTION CODE
10544  045624  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
10545  045630  000240                          NOP
10546  045632                         15$:
10547                          ;REPORT AN ERROR IF COMPOSITE ERROR IS SET AND NO OTHER
10548                          ;ERRORS ARE SET, OR IF COMPOSITE ERROR IS NOT SET AND
10549                          ;OTHER ERRORS ARE SET
10550  045632  005037  001140                  CLR     $GDDAT          ;EXPECT "ERR"=0
10551  045636  005737  001344                  TST     RMER1I          ;IS RMER1 =0??
10552  045642  001003                          BNE     20$             ;NO!!
10553  045644  005737  001372                  TST     RMER2I          ;IS RMERZ=0??
10554  045650  001403                          BEQ     25$             ;YES!!
10555  045652  052737  040000  001140  20$:     BIS     #ERR,$GDDAT     ;"ERR" SOULD BE SET
10556  045660  013737  001342  001142  25$:     MOV     RMDSI,$BDDAT
10557  045666  042737  137777  001142          BIC     #^CERR,$BDDAT
10558  045674  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "ERR" OK??
10559  045702  001412                          BEQ     30$             ;YES!!
10560  045704  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
10561  045710  112776  000047  000000          MOVB    #47,@(SP)       ;WRITE ERROR NUMBER
10562  045716  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10563  045722  004736                          JSR     PC,@(SP)+       ;REPORT INVALID COMP ERROR
10564  045724  162716  000010                  SUB     #10,(SP)
10565
10566                          ;REPORT AN ERROR IF "TRE" IS SET AND NONE OF THE BITS WHICH SET
10567                          ;TRE IS SET, OR IF TRE IS NOT SET AND ONE OR MORE BITS WHICH
10568                          ;SET TRE IS SET
10569  045730  005037  001140  30$:     CLR     $GDDAT          ;EXPECT "TRE" =0
10570  045734  013746  001340          MOV     RMCS2I,-(SP)    ;WAS DLT, WCE, UPE, NED, NEM
10571  045740  042726  000377          BIC     #377,(SP)+      ;PGE, MXF OR MDPE SET
10572  045744  001010                  BNE     35$             ;YES!!
10573  045746  032737  040000  001342  BIT     #ERR,RMDSI      ;WAS EXCEPTION RECEIVED??
10574  045754  001407                  BEQ     40$             ;NO!!
10575  045756  022737  000030  051224  CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
10576  045764  103003                  BHIS    40$             ;NO!!
10577  045766  052737  040000  001140  35$:     BIS     #TRE,$GDDAT     ;"TRE" SHOULD BE SET
10578  045774  013737  001330  001142  40$:     MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
10579  046002  042737  137777  001142          BIC     #^CTRE,$BDDAT
10580  046010  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "TRE" OK??
10581  046016  001413                          BEQ     45$             ;YES!!
10582  046020  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10583  046024  112776  000014  000000          MOVB    #14,@(SP)       ;WRITE ERROR NUMBER
```

```
10584  046032  162716  000002          SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10585  046036  004736                  JSR     PC,@(SP)+        ;REPORT TRE ERROR
10586  046040  162716  000010          SUB     #10,(SP)         ;RESTORE (SP)
10587  046044  000240                  NOP
10588  046046                       45$:
10589
10590                                ;;*****************************************************************
10591                                ;USING THE FUNCTION CODE TABLE, CHECK FOR THE FOLLOWING ERRORS:
10592                                ;       .STATUS BITS NOT SET THAT SHOULD BE SET, E.G., ATA AND ILF
10593                                ;       .STATUS BITS SET THAT SHOULD NOT BE SET, E.G., WCE AND ECH
10594                                ;NOTE THAT SOME ERROR BITS ARE CONDITIONAL ON THE COMMAND AND OTHER
10595                                ;STATUS CONDITIONS, E.G., WRITE LOCK ERROR SHOULD ONLY BE SET IF
10596                                ;WRITE LOCK IS ON AND THE COMMAND IS A WRITE.
10597
10598                                ;GET AND STORE THE ENTRY FROM THE FUNCTION CODE TABLE
10599  046046  010046                  MOV     R0,-(SP)         ;;PUSH R0 ON STACK
10600  046050  013700  051224          MOV     $15$,R0          ;R0 = FUNCTION CODE
10601  046054  016037  070676  051216  MOV     FNCDTB(R0),500$  ;STORE ENTRY
10602  046062  012600                  MOV     (SP)+,R0         ;;POP STACK INTO R0
10603
10604                                ;REPORT AN ERROR IF AN UNEXPECTED ATTENTION OCCURRED OR IF
10605                                ;ATA IS NOT SET AND SHOULD BE SET.
10606  046064  013737  051216  001140  MOV     500$,$GDDAT      ;GET EXPECTED ATA STATUS
10607  046072  032737  040000  001342  BIT     #ERR,RMDSI       ;IS COMPOSITE ERROR SET ??
10608  046100  001403                  BEQ     50$              ;NO !!
10609  046102  052737  100000  001140  BIS     #ATA,$GDDAT      ;EXPECT AN ATTENTION
10610  046110  042737  077777  001140  50$:    BIC     #^CATA,$GDDAT    ;STRIP DONT CARES
10611  046116  013737  001342  001142  MOV     RMDSI,$BDDAT     ;GET RECEIVED ATA
10612  046124  042737  077777  001142  BIC     #^CATA,$BDDAT    ;STRIP DONT CARES
10613  046132  023737  001140  001142  CMP     $GDDAT,$BDDAT    ;IS ATA OK ??
10614  046140  001413                  BEQ     55$              ;YES !!
10615  046142  062716  000004          ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
10616  046146  112776  000006  000000  MOVB    #6,@(SP)         ;LOAD ERROR # IN CALL
10617  046154  162716  000002          SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
10618  046160  004736                  JSR     PC,@(SP)+        ;REPORT ERROR
10619  046162  162716  000010          SUB     #10,(SP)         ;RESTORE SP
10620  046166  000240                  NOP
10621  046170                       55$:
10622
10623                                ;REPORT ERROR IF ILF IS INCORRECT, I.E., IF ILF DOES NOT COMPARE
10624                                ;WITH FUNCTION CODE TABLE
10625  046170  013737  051216  001140  MOV     500$,$GDDAT      ;GET EXPECTED ILF
10626  046176  042737  177776  001140  BIC     #^CILF,$GDDAT    ;CLEAR ALL OTHER BITS
10627  046204  013737  001344  001142  MOV     RMER1I,$BDDAT    ;GET RECEIVED ILF
10628  046212  042737  177776  001142  BIC     #^CILF,$BDDAT    ;CLEAR ALL OTHER BITS
10629  046220  023737  001140  001142  CMP     $GDDAT,$BDDAT    ;IS ILF OK ??
10630  046226  001412                  BEQ     60$              ;YES !!
10631  046230  062716  000004          ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
10632  046234  112776  000254  000000  MOVB    #254,@(SP)       ;WRITE ERROR NUMBER IN CALL
10633  046242  162716  000002          SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
10634  046246  004736                  JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
10635  046250  162716  000010          SUB     #10,(SP)         ;MOVE SP TO NO ERROR
10636  046254  005037  001140       60$:    CLR     $GDDAT           ;CLEAR EXPECTED STATUS
10637
10638                                ;REPORT AN ERROR IF WCE IS SET AND SHOULD NOT BE SET
10639  046260  013746  051216          MOV     500$,-(SP)       ;GET WCE STATUS ENABLE
```

```
10640  046264  052716  137777                  BIS     #↑CWCE,(SP)     ;SET ALL OTHER BITS
10641  046270  013737  001340  001142          MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
10642  046276  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR WCE IF ENABLED
10643  046302  001412                          BEQ     90$             ;BRANCH IF WCE OK
10644  046304  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10645  046310  112776  000026  000000          MOVB    #26,@(SP)       ;WRITE ERROR NUMBER
10646  046316  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10647  046322  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
10648  046324  162716  000010                  SUB     #10,(SP)        ;RESTORE ERROR
10649  046330                          90$:
10650
10651                                  ;REPORT ERROR IF OPI STATUS IS SET AND SHOULD NOT BE SET
10652  046330  013746  051216                  MOV     500$,-(SP)      ;GET OPI STATUS ENABLE
10653  046334  052716  157777                  BIS     #↑COPI,(SP)     ;SET ALL OTHER BITS
10654  046340  013737  001344  001142          MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10655  046346  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR OPI IF ENABLED
10656  046352  001412                          BEQ     100$            ;BRANCH IF OPI OK
10657  046354  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10658  046360  112776  000164  000000          MOVB    #164,@(SP)      ;WRITE ERROR NUMBER IN CALL
10659  046366  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10660  046372  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
10661  046374  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
10662  046400                          100$:
10663
10664                                  ;REPORT ERROR IF IVC IS SET AND IS NOT ENABLED OR IF IVC IS
10665                                  ;SET AND VV IS NOT RESET
10666  046400  013746  051216                  MOV     500$,-(SP)      ;GET IVC STATUS ENABLE
10667  046404  032737  000100  001342          BIT     #VV,RMDSI       ;IS VV SET
10668  046412  001402                          BEQ     105$            ;NO !!
10669  046414  042716  010000                  BIC     #IVC,(SP)       ;YES - IVC SHOULD BE 0
10670  046420  052716  167777          105$:   BIS     #↑CIVC,(SP)     ;SET ALL OTHER BITS
10671  046424  013737  001372  001142          MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
10672  046432  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR IVC IF ENABLED
10673  046436  001412                          BEQ     110$            ;BRANCH IF IVC OK
10674  046440  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10675  046444  112776  000165  000000          MOVB    #165,@(SP)      ;WRITE ERROR NUMBER IN CALL
10676  046452  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10677  046456  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
10678  046460  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10679  046464                          110$:
10680
10681                                  ;BIT 11 (WLE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10682                                  ;  ALL WRITE ERRORS, I.E.,
10683                                  ;       RMER1 - WLE, WCF
10684                                  ;       RMER2 - DPE
10685                                  ;       RMCS2 - UPE.
10686                                  ;EACH OF THESE ERRORS IS CHECKED TO SEE IF AN ERROR IS SET WHEN THE
10687                                  ;WRITE ERROR ENABLE BIT IS RESET.
10688
10689                                  ;REPORT AN ERROR IF WLE IS SET AND WRITE ERRORS ARE NOT ENABLED, OR IF
10690                                  ;THE DRIVE IS NOT WRITE PROTECTED
10691  046464  012746  177777                  MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ENABLED
10692  046470  032737  004000  051216          BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
10693  046476  001404                          BEQ     115$            ;NO !!
10694  046500  032737  004000  001342          BIT     #WRL,RMDSI      ;IS THE DRIVE WRITE PROTECTED ??
10695  046506  001002                          BNE     120$            ;YES !!
```

```
10696  046510  042716  004000          115$:   BIC     #WLE,(SP)          ;RESET WLE ENABLE
10697  046514  013737  001344  001142  120$:   MOV     RMER1I,$BDDAT      ;GET RECEIVED STATUS
10698  046522  042637  001142                  BIC     (SP)+,$BDDAT       ;CLEAR WLE IF ENABLED
10699  046526  001412                          BEQ     125$               ;BRANCH IF WLE OK
10700  046530  062716  000004                  ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
10701  046534  112776  000023  000000          MOVB    #23,@(SP)          ;WRITE ERROR NUMBER IN CALL
10702  046542  162716  000002                  SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
10703  046546  004736                          JSR     PC,@(SP)+          ;REPORT ERROR AND RETURN
10704  046550  162716  000010                  SUB     #10,(SP)           ;RESTORE SP TO NO ERROR
10705  046554                          125$:
10706
10707                                  ;REPORT ERROR IF WCF IS SET AND WRITE ERRORS ARE NOT ENABLED
10708  046554  012746  177777                  MOV     #-1,-(SP)          ;ASSUME WRITE ERRORS ENABLED
10709  046560  032737  004000  051216          BIT     #WLE,500$          ;ARE WRITE ERRORS ENABLED ??
10710  046566  001002                          BNE     130$               ;YES !!
10711  046570  042716  000040                  BIC     #WCF,(SP)          ;DISABLE WCF ERROR
10712  046574  013737  001344  001142  130$:   MOV     RMER1I,$BDDAT      ;GET RECEIVED STATUS
10713  046602  042637  001142                  BIC     (SP)+,$BDDAT       ;RESET WCF IF ENABLED
10714  046606  001412                          BEQ     135$               ;BRANCH IF WCF OK
10715  046610  062716  000004                  ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
10716  046614  112776  000025  000000          MOVB    #25,@(SP)          ;WRITE ERROR NUMBER IN CALL
10717  046622  162716  000002                  SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
10718  046626  004736                          JSR     PC,@(SP)+          ;REPORT ERROR
10719  046630  162716  000010                  SUB     #10,(SP)           ;RESTORE SP TO NO ERROR
10720  046634                          135$:
10721
10722                                  ;REPORT ERROR IF DPE IS SET AND WRITE ERRORS ARE NOT ENABLED
10723  046634  012746  177777                  MOV     #-1,-(SP)          ;ASSUME WRITE ERRORS ARE ENABLED
10724  046640  032737  004000  051216          BIT     #WLE,500$          ;ARE WRITE ERRORS ENABLED ??
10725  046646  001002                          BNE     140$               ;YES !!
10726  046650  042716  000010                  BIC     #DPE,(SP)          ;RESET DPE ENABLE
10727  046654  013737  001372  001142  140$:   MOV     RMER2I,$BDDAT      ;GET RECEIVED STATUS
10728  046662  042637  001142                  BIC     (SP)+,$BDDAT       ;RESET DPE IF ENABLED
10729  046666  001412                          BEQ     145$               ;BRANCH IF DPE OK
10730  046670  062716  000004                  ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
10731  046674  112776  000040  000000          MOVB    #40,@(SP)          ;WRITE ERROR NUMBER IN CALL
10732  046702  162716  000002                  SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
10733  046706  004736                          JSR     PC,@(SP)+          ;REPORT ERROR
10734  046710  162716  000010                  SUB     #10,(SP)           ;RESTORE SP TO NO ERROR
10735  046714                          145$:
10736
10737                                  ;REPORT AN ERROR IF UPE IS SET AND WRITE ERRORS ARE NOT ENABLED
10738  046714  012746  177777                  MOV     #-1,-(SP)          ;ASSUME WRITE ERRORS ARE ENABLED
10739  046720  032737  004000  051216          BIT     #WLE,500$          ;ARE WRITE ERRORS ENABLED ??
10740  046726  001002                          BNE     150$               ;YES !!
10741  046730  042716  020000                  BIC     #UPE,(SP)          ;DISABLE UPE ERROR
10742  046734  013737  001340  001142  150$:   MOV     RMCS2I,$BDDAT      ;GET RECEIVED STATUS
10743  046742  042637  001142                  BIC     (SP)+,$BDDAT       ;RESET UPE IF ENABLED
10744  046746  001412                          BEQ     155$               ;BRANCH IF UPE OK
10745  046750  062716  000004                  ADD     #4,(SP)            ;MOVE SP TO USERS ERROR CALL
10746  046754  112776  000024  000000          MOVB    #24,@(SP)          ;WRITE ERROR NUMBER IN CALL
10747  046762  162716  000002                  SUB     #2,(SP)            ;MOVE SP TO ERROR RETURN
10748  046766  004736                          JSR     PC,@(SP)+          ;REPORT ERROR AND RETURN
10749  046770  162716  000010                  SUB     #10,(SP)           ;MOVE SP TO NO ERROR
10750  046774                          155$:
10751
```

```
10752
10753  046774  013746  051216                  ;REPORT AN ERROR IF IAE IS SET AND IS NOT ENABLED
10754  047000  052716  175777                          MOV     500$,-(SP)      ;GET IAE ENABLE
10755  047004  013737  001344  001142                  BIS     #^CIAE,(SP)     ;SET ALL OTHER BITS
10756  047012  042637  001142                          MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10757  047016  001412                                  BIC     (SP)+,$BDDAT    ;CLEAR IAE IF ENABLED
10758  047020  062716  000004                          BEQ     160$            ;BRANCH IF IAE IS OK
10759  047024  112776  000166  000000                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10760  047032  162716  000002                          MOVB    #166,@(SP)      ;WRITE ERROR NUMBER
10761  047036  004736                                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10762  047040  162716  000010                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10763  047044                                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
                                               160$:
10764
10765                                          ;BIT 09 (AOE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10766                                          ; ALL READ/WRITE ERRORS, I.E.,
10767                                          ;       RMCS1 - TRE
10768                                          ;       RMCS2 - DLT,NEM,MXF
10769                                          ;       RMDS  - LBT
10770                                          ;       RMER1 - AOE
10771                                          ;NOTE:
10772                                          ;       LBT IS NOT CHECKED BECAUSE IT ONLY RESETS WHEN THE DESIRED
10773                                          ;CYLINDER REGISTER IS WRITTEN
10774                                          ;NOTE:
10775                                          ;       AOE CANNOT BE SET IF LBT IS NOT ALSO SET
10776                                          ;NOTE:
10777                                          ;       TRE IS CHECKED AS A FUNCTION OF OTHER ERROR CONDITONS ABOVE
10778
10779
10780                                          ;REPORT AN ERROR IF DLT IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10781  047044  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10782  047050  032737  001000  051216                  BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
10783  047056  001002                                  BNE     165$            ;YES !!
10784  047060  042716  100000                          BIC     #DLT,(SP)       ;RESET DLT ENABLE
10785  047064  013737  001340  001142          165$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10786  047072  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR DLT IF ENABLED
10787  047076  001412                                  BEQ     170$            ;BRANCH IF DLT IS OK
10788  047100  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10789  047104  112776  000032  000000                  MOVB    #32,@(SP)       ;WRITE ERROR NUMBER IN CALL
10790  047112  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10791  047116  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10792  047120  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10793  047124                                 170$:
10794
10795                                          ;REPORT ERROR IF NEM IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10796  047124  012746  177777                          MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10797  047130  032737  001000  051216                  BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
10798  047136  001002                                  BNE     175$            ;YES !!
10799  047140  042716  004000                          BIC     #NEM,(SP)       ;DISABLE NEM
10800  047144  013737  001340  001142          175$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10801  047152  042637  001142                          BIC     (SP)+,$BDDAT    ;CLEAR NEM IF ENABLED
10802  047156  001412                                  BEQ     180$            ;BRANCH IF NEM IS OK
10803  047160  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10804  047164  112776  000167  000000                  MOVB    #167,@(SP)      ;WRITE ERROR NUMBER IN CALL
10805  047172  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10806  047176  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10807  047200  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
```

```
10808  047204                          180$:
10809
10810                                  ;REPORT ERROR IF MXF IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
10811  047204  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
10812  047210  032737  001000  051216        BIT     #AOE,500$       ;ARE DATA ERRORS ENABLED ??
10813  047216  001002                        BNE     185$            ;YES !!
10814  047220  042716  001000                BIC     #MXF,(SP)       ;DISABLE MXF ERROR
10815  047224  013737  001340  001142 185$:  MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10816  047232  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR MXF IF ENABLED
10817  047236  001412                        BEQ     190$            ;BRANCH IF MXF IS OK
10818  047240  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10819  047244  112776  000033  000000        MOVB    #33,@(SP)       ;WRITE ERROR NUMBER IN CALL
10820  047252  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10821  047256  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10822  047260  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10823  047264                          190$:
10824
10825                                  ;REPORT ERROR IF AOE IS SET AND DATA ERRORS ARE NOT ENABLED
10826  047264  012746  177777                MOV     #-1,-(SP)       ;ASSUME DATA ERRORS ARE ENABLED
10827  047270  032737  001000  051216        BIT     #AOE,500$       ;ARE DATA ERRORS EANBLED ??
10828  047276  001404                        BEQ     191$            ;NO !!
10829  047300  032737  002000  001342        BIT     #LBT,RMDSI      ;IS LBT ALSO SET ??
10830  047306  001002                        BNE     195$            ;YES !!
10831  047310  042716  001000          191$: BIC     #AOE,(SP)       ;DISABLE AOE
10832  047314  013737  001344  001142 195$:  MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10833  047322  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR AOE IF ENABLED
10834  047326  001412                        BEQ     200$            ;BRANCH IF AOE IS OK
10835  047330  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10836  047334  112776  000020  000000        MOVB    #20,@(SP)       ;WRITE ERROR NUMBER
10837  047342  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10838  047346  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10839  047350  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10840  047354                          200$:
10841
10842                                  ;BIT 07 (HCE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
10843                                  ;HEADER ERRORS, I.E.,
10844                                  ;       RMER1 - HCRC,HCE,FER
10845                                  ;       RMER2 - BSE
10846
10847                                  ;RESET THE ENABLING BIT (HCE) IF HEADER COMPARE INHIBIT IS SET
10848  047354  032737  002000  001362        BIT     #HCI,RMOFI      ;IS HCI SET ??
10849  047362  001403                        BEQ     201$            ;NO !!
10850  047364  042737  000200  051216        BIC     #HCE,500$       ;YES - DISABLE ALL HEADER ERRORS
10851  047372                          201$:
10852
10853                                  ;REPORT AN ERROR IF HCRC IS SET AND HEADER ERRORS ARE NOT ENABLED
10854  047372  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10855  047376  032737  000200  051216        BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
10856  047404  001002                        BNE     205$            ;YES !!
10857  047406  042716  000400                BIC     #HCRC,(SP)      ;DISABLE HCRC
10858  047412  013737  001344  001142 205$:  MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10859  047420  042637  001142                BIC     (SP)+,$BDDAT    ;RESET HCRC IF ENABLED
10860  047424  001412                        BEQ     210$            ;BRANCH IF HCRC IS OK
10861  047426  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10862  047432  112776  000035  000000        MOVB    #35,@(SP)       ;WRITE ERROR NUMBER IN CALL
10863  047440  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
```

```
10864  047444  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10865  047446  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10866  047452                          210$:
10867
10868                                  ;REPORT ERROR IF HCE IS SET AND HEADER ERRORS ARE NOT ENABLED
10869  047452  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10870  047456  032737  000200  051216          BIT     #HCE,S00$       ;ARE ERRORS ENABLED ??
10871  047464  001002                          BNE     215$            ;YES !!
10872  047466  042716  000200                  BIC     #HCE,(SP)       ;DISABLE HCE
10873  047472  013737  001344  001142  215$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10874  047500  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR HCE IF ENABLED
10875  047504  001412                          BEQ     220$            ;BRANCH IF HCE IS OK
10876  047506  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10877  047512  112776  000036  000000          MOVB    #36,@(SP)       ;WRITE ERROR NUMBER IN CALL
10878  047520  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10879  047524  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10880  047526  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10881  047532                          220$:
10882
10883                                  ;REPORT ERROR IF FER IS SET AND HEADER ERRORS ARE NOT ENABLED
10884  047532  012746  177777                  MOV     #-1,-(SP)       ;ASSUME FER IS ENABLED
10885  047536  032737  000200  051216          BIT     #HCE,S00$       ;ARE HEADER ERRORS ENABLED ??
10886  047544  001002                          BNE     225$            ;YES !!
10887  047546  042716  000020                  BIC     #FER,(SP)       ;DISABLE FER
10888  047552  013737  001344  001142  225$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10889  047560  042637  001142                  BIC     (SP)+,$BDDAT    ;RESET FER IF ENABLED
10890  047564  001412                          BEQ     230$            ;BRANCH IF FER OK
10891  047566  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10892  047572  112776  000037  000000          MOVB    #37,@(SP)       ;WRITE ERROR NUMBER IN CALL
10893  047600  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10894  047604  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10895  047606  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10896  047612                          230$:
10897
10898                                  ;REPORT ERROR IF BSE IS SET AND HEADER ERRORS ARE NOT ENABLED
10899  047612  012746  177777                  MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
10900  047616  032737  000200  051216          BIT     #HCE,S00$       ;ARE THEY ENABLED ??
10901  047624  001002                          BNE     235$            ;YES !!
10902  047626  042716  100000                  BIC     #BSE,(SP)       ;DISABLE BSE
10903  047632  013737  001372  001142  235$:   MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
10904  047640  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR BSE IF ENABLED
10905  047644  001412                          BEQ     240$            ;BRANCH IF BSE OK
10906  047646  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10907  047652  112776  000354  000000          MOVB    #354,@(SP)      ;WRITE ERROR NUMBER
10908  047660  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10909  047664  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10910  047666  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10911  047672                          240$:
10912
10913                                  ;BIT 06 OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR DATA
10914                                  ;FIELD ERRORS, I.E.,
10915                                  ;       RMCS2 - MDPE
10916                                  ;       RMER1 - DCK,ECH
10917                                  ;NOTE:
10918                                  ;       ECH CANNOT SET UNLESS IT IS ENABLED AND ECI IS RESET AND
10919                                  ;DCK IS SET.
```

```
10920
10921                                   ;REPORT ERROR IF MDPE IS SET AND IS NOT ENABLED
10922  047672  012746  177777                 MOV     #-1,-(SP)       ;ASSUME ENABLED
10923  047676  032737  000100  051216         BIT     #ECH,500$       ;ARE DATA FIELD ERRORS ENABLED ??
10924  047704  001002                         BNE     245$            ;YES !!
10925  047706  042716  000400                 BIC     #MDPE,(SP)      ;DISBALE MDPE
10926  047712  013737  001340  001142 245$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
10927  047720  042637  001142                 BIC     (SP)+,$BDDAT    ;CLEAR MDPE IF ENABLED
10928  047724  001412                         BEQ     250$            ;BRANCH IF MDPE OK
10929  047726  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10930  047732  112776  000027  000000         MOVB    #27,@(SP)       ;WRITE ERROR NUMBER IN CALL
10931  047740  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10932  047744  004736                         JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10933  047746  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10934  047752                          250$:
10935
10936                                   ;REPORT ERROR IF DCK IS SET AND DATA FIELD ERRORS ARE NOT ENABLED
10937  047752  012746  177777                 MOV     #-1,-(SP)       ;ASSUME ENABLED
10938  047756  032737  000100  051216         BIT     #ECH,500$       ;ARE THEY ENABLED ??
10939  047764  001002                         BNE     255$            ;YES !!
10940  047766  042716  100000                 BIC     #DCK,(SP)       ;DISABLE DCK
10941  047772  013737  001344  001142 255$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10942  050000  042637  001142                 BIC     (SP)+,$BDDAT    ;CLEAR DCK IF ENABLED
10943  050004  001412                         BEQ     260$            ;BRANCH IF DCK IS OK
10944  050006  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10945  050012  112776  000030  000000         MOVB    #30,@(SP)       ;WRITE ERROR NUMBER IN CALL
10946  050020  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10947  050024  004736                         JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10948  050026  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10949  050032                          260$:
10950
10951                                   ;REPORT ERROR IF ECH IS SET AND,
10952                                   ;       DATA FIELD ERRORS ARE NOT ENABLED, OR
10953                                   ;       ECI IS SET, OR
10954                                   ;       DCK IS NOT SET,
10955  050032  012746  177777                 MOV     #-1,-(SP)       ;ASSUME ENABLED
10956  050036  032737  000100  051216         BIT     #ECH,500$       ;ARE ERRORS ENABLED ??
10957  050044  001410                         BEQ     265$            ;NO !!
10958  050046  032737  004000  001362         BIT     #ECI,RMOFI      ;IS ECI SET ??
10959  050054  001004                         BNE     265$            ;YES !!
10960  050056  032737  100000  001344         BIT     #DCK,RMER1I     ;IS DCK ALSO SET ??
10961  050064  001002                         BNE     270$            ;YES !!
10962  050066  042716  000100          265$:  BIC     #ECH,(SP)       ;DISABLE ECH
10963  050072  013737  001344  001142 270$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10964  050100  042637  001142                 BIC     (SP)+,$BDDAT    ;CLEAR ECH IF ENABLED
10965  050104  001412                         BEQ     275$            ;BRANCH IF ECH IS OK
10966  050106  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10967  050112  112776  000031  000000         MOVB    #31,@(SP)       ;WRITE ERROR NUMBER IN CALL
10968  050120  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
10969  050124  004736                         JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10970  050126  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10971  050132                          275$:
10972
10973                                   ;;*****************************************************************
10974                                   ;PERFORM THE REMAINING ERROR CHECKS ONLY FOR DATA TRANSFER COMMANDS
10975
```

```
10976  050132  022737  000030  051224          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
10977  050140  103402                          BLO     280$            ;YES!!
10978  050142  000137  051170                  JMP     355$
10979
```

```
10980                                        ;THE FOLLOWING STATUS CHECKS ARE FOR DATA TRANSFER COMMANDS ONLY
10981                                        ;REPORT ERROR IF RMWC NOT ZERO AND TRE IS ZERO
10982  050146  013737  001332  001142  280$:  MOV     RMWCI,$BDDAT     ;WORD COUNT ZERO??
10983  050154  001421                       BEQ     285$             ;YES
10984  050156  032737  040000  001330       BIT     #TRE,RMCS1I      ;TRANSFER ERROR DETECTED??
10985  050164  001015                       BNE     285$             ;YES!!
10986  050166  062716  000004               ADD     #4,(SP)
10987  050172  112776  000015  000000       MOVB    #15,@(SP)        ;ERROR NUMBER
10988  050200  005037  001140               CLR     $GDDAT           ;GOOD DADA FOR TYPEOUT
10989  050204  162716  000002               SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
10990  050210  004736                       JSR     PC,@(SP)+        ;REPORT WORD COUNT NOT ZERO
10991  050212  162716  000010               SUB     #10,(SP)         ;RESTORE (SP)
10992  050216  000240                       NOP
10993                                        ;REPORT ERROR IF RMBA IS NOT CORRECT
10994  050220  013737  001332  001140  285$:  MOV     RMWCI,$GDDAT     ;NUMBER OF WORDS TRANSFERRED
10995  050226  163737  001402  001140       SUB     RMWCO,$GDDAT
10996  050234  006337  001140               ASL     $GDDAT
10997  050240  063737  001404  001140       ADD     RMBAO,$GDDAT     ;EXPECTED BUS ADDRESS
10998  050246  032737  000010  001340       BIT     #BAI,RMCS2I      ;WAS BAI SET ??
10999  050254  001403                       BEQ     290$             ;NO !!
11000  050256  013737  001404  001140       MOV     RMBAO,$GDDAT     ;ADDRESS SHOULD NOT HAVE CHANGED
11001  050264  023737  001140  001334  290$:  CMP     $GDDAT,RMBAI     ;BUS ADDRESS OK??
11002  050272  001416                       BEQ     295$             ;YES!!
11003  050274  013737  001334  001142       MOV     RMBAI,$BDDAT     ;BAD DATA FOR TYPEOUT
11004  050302  062716  000004               ADD     #4,(SP)
11005  050306  112776  000016  000000       MOVB    #16,@(SP)        ;ERROR NUMBER
11006  050314  162716  000002               SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11007  050320  004736                       JSR     PC,@(SP)+        ;REPORT UNEXPECTED ADDRESS
11008  050322  162716  000010               SUB     #10,(SP)                 ;RESTORE (SP)
11009  050326  000240                       NOP
11010                                        ;COMPUTE NUMBER OF SECTORS TRANSFERRED
11011  050330  005046                  295$:  CLR     -(SP)            ;NUMBER OF SECTORS TRANSFERRED
11012  050332  013746  001332               MOV     RMWCI,-(SP)      ;NUMBER OF WORDS TRANSFERRED
11013  050336  163716  001402               SUB     RMWCO,(SP)
11014  050342  012746  000400               MOV     #256.,-(SP)      ;NUMBER OF WORDS PER SECTOR
11015  050346  032737  000002  001400       BIT     #BIT1,RMCS1O     ;HEADER & DATA COMMAND ??
11016  050354  001402                       BEQ     300$             ;NO !!
11017  050356  012716  000402               MOV     #258.,(SP)       ;YES - CHANGE WORDS PER SECTOR
11018  050362  005266  000004          300$:  INC     4(SP)            ;INCREMENT SECTOR COUNT
11019  050366  161666  000002               SUB     (SP),2(SP)       ;SUBTRACT ONE SECTOR'S WORTH
11020  050372  003373                       BGT     300$             ;CONTINUE IF NOT DONE
11021  050374  022626                       CMP     (SP)+,(SP)+      ;STRIP 2 FROM STACK
11022                                        ;COMPUTE EXPECTED SECTOR, TRACK AND CYLINDER ADDRESS FROM NUMBER OF SECTORS
11023  050376  013737  001406  051222       MOV     RMDAO,510$       ;STORE ORIGINAL SECTOR
11024  050404  013737  001406  051220       MOV     RMDAO,505$       ;STORE ORIGINAL TRACK
11025  050412  013737  001434  051216       MOV     RMDCO,500$       ;STORE ORIGINAL CYLINDER
11026  050420  042737  177740  051222       BIC     #^C37,510$
11027  050426  000337  051220               SWAB    505$
11028  050432  042737  177770  051220       BIC     #^C7,505$
11029  050440  062637  051222               ADD     (SP)+,510$
11030
11031  050444  023727  051222  000040  305$:  CMP     510$,#32.        ;SECTOR OVEFLOWED??
11032  050452  103420                       BLO     315$             ;NO!!
11033  050454  023727  051222  000240       CMP     510$,#<5*32.>    ;CYLINDERS WORTH??
11034  050462  103406                       BLO     310$             ;NO!!
11035  050464  005237  051216               INC     500$             ;YES INCREMENT CYLINDER
```

```
11036  050470  162737  000240  051222          SUB     #<5*32.>,510$   ;ADJUST SECTOR
11037  050476  000762                          BR      305$            ;TRY AGAIN
11038  050500  005237  051220          310$:   INC     505$            ;INCREMENT TRACK
11039  050504  162737  000040  051222          SUB     #32.,510$       ;ADJUST SECTOR
11040  050512  000754                          BR      305$            ;TRY AGAIN
11041
11042  050514  023727  051220  000005  315$:   CMP     505$,#5 ;TRACK OVERFLOWED??
11043  050522  103406                          BLO     320$            ;NO!!
11044  050524  005237  051216                  INC     500$            ;INCREMENT CYLINDER
11045  050530  162737  000005  051220          SUB     #5,505$ ;ADJUST TRACK
11046  050536  000766                          BR      315$            ;TRY AGAIN
11047                          ;REPORT ERROR IF "LBT" IS NOT CORRECT
11048  050540  005037  001140          320$:   CLR     $GDDAT          ;SET GOOD DATA FOR LBT=0
11049  050544  022737  001467  051216          CMP     #823.,500$      ;SHOULD LBT BE SET??
11050  050552  101007                          BHI     325$            ;NO!!
11051  050554  032737  002000  001344          BIT     #IAE,RMER1I     ;WAS IAE SET ??
11052  050562  001003                          BNE     325$            ;YES - LBT SHOULD NOT BE SET
11053  050564  012737  002000  001140          MOV     #LBT,$GDDAT     ;SET GOOD DATA FOR LBT=1
11054  050572  013737  001342  001142  325$:   MOV     RMDS1,$BDDAT    ;BAD DATA FOR TYPEOUT
11055  050600  042737  175777  001142          BIC     #^CLBT,$BDDAT
11056  050606  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS LBT CORRECT??
11057  050614  001413                          BEQ     330$            ;YES!!
11058  050616  062716  000004                  ADD     #4,(SP)
11059  050622  112776  000017  000000          MOVB    #17,@(SP)       ;ERROR NUMBER
11060  050630  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11061  050634  004736                          JSR     PC,@(SP)+       ;REPORT LBT IS WRONG
11062  050636  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11063  050642  000240                          NOP
11064                          ;REPORT ERROR IF "AOE" IS INCORRECT
11065  050644  005037  001140          330$:   CLR     $GDDAT          ;SET FOR AOE=0
11066  050650  032737  002000  001344          BIT     #IAE,RMER1I     ;WAS "IAE" DETECTED??
11067  050656  001031                          BNE     340$            ;YES-"AOE" SHOULD BE ZERO
11068  050660  022737  001467  051216          CMP     #823.,500$      ;SHOULD AOE BE SET??
11069  050666  101025                          BHI     340$            ;NO!!
11070  050670  005737  051220                  TST     505$            ;MAYBE
11071  050674  001012                          BNE     335$            ;YES
11072  050676  005737  051222                  TST     510$
11073  050702  001007                          BNE     335$            ;YES !!
11074  050704  032737  000010  051224          BIT     #F2,515$        ;WAS THIS READ OR WRITE CHECK ??
11075  050712  001413                          BEQ     340$            ;NO !!
11076  050714  005737  001332                  TST     RMWCI           ;WAS ALL DATA TRANSFERRED ??
11077  050720  001410                          BEQ     340$            ;YES !!
11078  050722  012737  001000  001140  335$:   MOV     #AOE,$GDDAT     ;SET FOR AOE=1
11079  050730  005037  051220                  CLR     505$            ;CLEAR EXPECTED TRACK
11080  050734  012737  000001  051222          MOV     #1,510$ ;EXPECT SECTOR=1
11081  050742  013737  001344  001142  340$:   MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
11082  050750  042737  176777  001142          BIC     #^CAOE,$BDDAT
11083  050756  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS AOE CORRECTY??
11084  050764  001413                          BEQ     345$            ;YES!!
11085  050766  062716  000004                  ADD     #4,(SP)
11086  050772  112776  000020  000000          MOVB    #20,@(SP)       ;ERROR NUMBER
11087  051000  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11088  051004  004736                          JSR     PC,@(SP)+       ;REPORT AOE IS WRONG
11089  051006  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11090  051012  000240                          NOP
11091                          ;REPORT ERROR IF RMDA IS NOT CORRECT
```

```
11092  051014  032737  002000  001344  345$:   BIT     #IAE,RMER1I     ;WAS THERE AN IAE ERROR ??
11093  051022  001062                          BNE     355$            ;YES - DONT CHECK RMDA,RMDC
11094  051024  013737  051220  001140          MOV     505$,$GDDAT     ;SETUP EXPECTED DISK ADDRESS
11095  051032  000337  001140                  SWAB    $GDDAT
11096  051036  113737  051222  001140          MOVB    510$,$GDDAT
11097  051044  013737  001336  001142          MOV     RMDAI,$BDDAT    ;SETUP RECEIVED DISK ADDRESS
11098  051052  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11099  051060  001413                          BEQ     350$            ;BRANCH IF EQUAL
11100  051062  062716  000004                  ADD     #4,(SP)
11101  051066  112776  000021  000000          MOVB    #21,@(SP)       ;ERROR NUMBER
11102  051074  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11103  051100  004736                          JSR     PC,@(SP)+       ;REPORT BAD DISK ADDRESS
11104  051102  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11105  051106  000240                          NOP
11106                          ;REPORT ERROR IF RMDC IS INCORRECT
11107  051110  013737  051216  001140  350$:   MOV     500$,$GDDAT     ;SETUP EXPECTED CYLINDER
11108  051116  042737  176000  001140          BIC     #^C1777,$GDDAT
11109  051124  013737  001364  001142          MOV     RMDCI,$BDDAT    ;SETUP RECEIVED CYLINDER
11110  051132  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE CYLINDERS
11111  051140  001413                          BEQ     355$            ;BRANCH IF EQUAL
11112  051142  062716  000004                  ADD     #4,(SP)
11113  051146  112776  000022  000000          MOVB    #22,@(SP)       ;ERROR NUMBER
11114  051154  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11115  051160  004736                          JSR     PC,@(SP)+       ;REPORT BAD CYLINDER
11116  051162  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11117  051166  000240                          NOP
```

```
11118  051170  062716  000004          355$:   ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
11119  051174  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
11120  051200  001403                          BEQ     360$
11121  051202  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR RETURN
11122  051206  000402                          BR      365$
11123  051210  162716  000004          360$:   SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
11124  051214  000207                  365$:   RTS     PC
11125
11126  051216  000000                  500$:   .WORD   0               ;CYLINDER
11127  051220  000000                  505$:   .WORD   0               ;TRACK
11128  051222  000000                  510$:   .WORD   0               ;SECTOR
11129  051224  000000                  515$:   .WORD   0               ;FUNCTION CODE
11130
11131
```

```
11132                                   .SBTTL  DEVICE SELECT SUBROUTINE
11133
11134                                   ; THIS SUBROUTINE SELECTS THE DEVICE, GETTING THE DEVICE NUMBER FROM THE
11135                                   ; TEST QUEUE.
11136
11137                                   ;CALL:
11138                                   ;(1)    JSR     PC,DEVSEL
11139                                   ;(2)    BR      ??              RETURN IF NO ERROR
11140                                   ;(3)    NOP                     RETURN IF ERROR
11141                                   ;(4)    ERROR                   ERROR DEFINED BY SUBROUTINE
11142
11143   051226                          DEVSEL:
11144
11145                                   ;CLEAR USER'S ERROR CALL
11146   051226  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
11147   051232  105076  000000                 CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF CALL
11148   051236  162716  000004                 SUB     #4,(SP)         ;MOVE SP BACK
11149                                   ;SAVE USER'S INFORMATION AND SETUP REGISTERS
11150   051242  013746  000004                 MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
11151   051246  013746  000006                 MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
11152   051252  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11153   051254  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11154   051256  012737  051376  000004         MOV     #20$,ERRVEC     ;SETUP FOR BUS TIMEOUT
11155   051264  012737  000300  000006         MOV     #PR6,ERRVEC+2
11156   051272  013700  001276                 MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
11157   051276  013701  001450                 MOV     TSTQUE,R1       ;R1 POINTS TO DEVICE NUMBER
11158
11159                                   ;SELECT DEVICE AND VERIFY THAT DEVICE IS AVAILABLE
11160   051302  111160  000010                 MOVB    (R1),RMCS2(R0)  ;WRITE UNIT SELECT BITS
11161   051306  016037  000000  001176         MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
11162   051314  016037  000010  001174         MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
11163   051322  032737  010000  001174.        BIT     #NED,$TMP0      ;IS DEVICE NONEXISTENT??
11164   051330  001407                         BEQ     10$             ;NO!!
11165   051332  062766  000004  000010         ADD     #4,10(SP)       ;MOVE SP TO USERS ERROR CALL
11166   051340  112776  000111  000010         MOVB    #111,@10(SP)    ;WRITE ERROR NUMBER
11167   051346  000422                         BR      30$             ;
11168   051350  032737  004000  001176 10$:    BIT     #DVA,$TMP1      ;IS DEVICE AVAILABLE??
11169   051356  001021                         BNE     35$             ;YES!!
11170   051360  062766  000004  000010         ADD     #4,10(SP)
11171   051366  112776  000112  000010         MOVB    #112,@10(SP)
11172   051374  000407                         BR      30$
11173
11174   051376                          20$:
11175
11176                                   ;HANDLE BUS TIMEOUT
11177   051376  022626                         CMP     (SP)+,(SP)+     ;ADJUST SP
11178   051400  062766  000004  000010         ADD     #4,10(SP)
11179   051406  112776  000113  000010         MOVB    #113,@10(SP)
11180   051414  162766  000002  000010 30$:    SUB     #2,10(SP)       ;MOVE SP TO RETURN IF ERROR
11181
11182   051422                          35$:
11183
11184
11185                                   ;RESTORE USERS DATA AND RETURN TO ADDRESS ON STACK
11186   051422  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
11187   051424  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
```

```
 11188  051426  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
 11189  051432  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 11190  051436  000207                        RTS     PC              ;EXIT
```

```
11191                                   .SBTTL  SEEK STATUS CHECK SUBROUTINE
11192                                   ;
11193                                   ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEEK TESTS USING STATUS
11194                                   ;STORED IN THE GET BUFFER AND TEST PARAMETERS STORED IN THE PUT BUFFER.
11195                                   ;
11196                                   ;
11197                                   ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS DETECTED
11198                                   ;AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN THE "ERROR" TRAP
11199                                   ;OF THE CALLING ROUTINE.  SEEK STATUS IS CHECKED AS FOLLOWS:
11200                                   ;
11201                                   ;CALL:
11202                                   ;(1)    JSR     PC,SEKSTS
11203                                   ;       BR      ???             RETURN HERE IF NO ERROR
11204                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
11205                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11206                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11207                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
11208
11209   051440                          SEKSTS:
11210
11211                                   ;CLEAR USER'S ERROR  CALL
11212   051440  000240                          NOP
11213   051442  062716  000004                ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
11214   051446  105076  000000                CLRB    @(SP)           ;CLEAR ERROR NUMBER
11215   051452  162716  000004                SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
11216   051456  005037  052676                CLR     300$            ;CLEAR ERROR FLAGS
11217
11218                                   ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING
11219                                   ;LOCAL REGISTERS, I.E., "PAR"  = 1 AND "DPE" = 0
11220   051462  032737  000010  001344          BIT     #PAR,RMER1I     ;WAS PARITY ERROR DETECTED??
11221   051470  001424                          BEQ     1$              ;NO!!
11222   051472  032737  000010  001372          BIT     #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
11223   051500  001020                          BNE     1$              ;NOT SURE!!
11224
11225                                   ;REPORT REGISTER PARITY ERROR VIA USER'S ERROR CALL
11226   051502  005037  001140                CLR     $GDDAT          ;EXPECTED STATUS
11227   051506  013737  001344  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11228   051514  062716  000004                ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
11229   051520  112776  000050  000000        MOVB    #50,@(SP)       ;ERROR #50
11230   051526  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11231   051532  004736                        JSR     PC,@(SP)+
11232   051534  162716  000010                SUB     #10,(SP)        ;RESTORE STACK
11233   051540  000437                        BR      3$              ;IAE SHOULD  BE ZERO
11234
11235
11236                                   ;DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK,SECTOR AND
11237                                   ;CYLINDER ADDRESS USED DURING SEEK OPERATION
11238   051542  012737  002000  001140  1$:     MOV     #IAE,$GDDAT     ;SET UP FOR IAE = 1
11239   051550  052737  040000  052676          BIS     #SKI,300$       ;SET SKI ERROR FLAG
11240   051556  023727  001434  001466          CMP     RMDC0,#822.     ;CYLINDER > 822??
11241   051564  101025                          BHI     3$              ;YES!!
11242   051566  042737  040000  052676          BIC     #SKI,300$       ;CLEAR SKI ERROR FLAG
11243   051574  123727  001406  000037          CMPB    RMDA0,#31.      ;SECTOR > 31??
11244   051602  101016                          BHI     3$              ;YES!!
11245   051604  123727  001406  000035          CMPB    RMDA0,#29.      ;SECTOR > 29 ??
11246   051612  101404                          BLOS    2$              ;NO!!
```

```
11247  051614  032737  010000  001432          BIT     #FMT16,RMOFO    ;30 SECTOR FORMAT??
11248  051622  001406                          BEQ     3$              ;YES!!
11249  051624  123727  001407  000004  2$:     CMPB    RMDA0+1,#4.     ;TRACK  >4??
11250  051632  101002                          BHI     3$              ;YES!!
11251  051634  005037  001140                  CLR     $GDDAT          ;"IAE" SHOULD BE 0
11252
11253                                  ;COMPARE EXPECTED AND RECEIVED "IAE" STATUS
11254  051640  013737  001344  001142  3$:     MOV     RMER1I,$BDDAT   ;IS IAE OK??
11255  051646  042737  175777  001142          BIC     #^CIAE,$BDDAT
11256  051654  023737  001140  001142          CMP     $GDDAT,$BDDAT
11257  051662  001004                          BNE     35$             ;IAE IN ERROR
11258  051664  042737  040000  052676          BIC     #SKI,300$       ;CLEAR SKI FLAG
11259  051672  000413                          BR      5$              ;GO CHECK NEXT ERROR
11260  051674                          35$:
11261                                  ;REPORT INCORRECT "IAE" STATUS VIA USER'S ERROR CALL
11262  051674  062716  000004                  ADD     #4,(SP)
11263  051700  112776  000051  000000          MOVB    #51,@(SP)       ;ERROR 51
11264  051706  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11265  051712  004736                          JSR     PC,@(SP)+       ;REPORT INCORRECT IAE
11266  051714  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
11267  051720  000240                          NOP
11268  051722                          5$:
11269
11270                                  ;REPORT ANY IVC ERROR AS
11271                                  ;       IVC ERROR WITH VOLUME VALID ZERO
11272                                  ;       ERRONEOUS IVC ERROR, VOLUME VALID IS SET
11273  051722  032737  010000  001372          BIT     #IVC,RMER2I     ;IVC ERROR??
11274  051730  001427                          BEQ     52$             ;NO!!
11275  051732  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
11276  051736  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11277  051744  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
11278  051750  112776  000060  000000          MOVB    #60,@(SP)       ;ERROR 60 IF VV = 0
11279  051756  032737  000100  001342          BIT     #VV,RMDSI
11280  051764  001403                          BEQ     51$
11281  051766  112776  000061  000000          MOVB    #61,@(SP)       ;ERROR 61 IF VV  = 1
11282  051774  162716  000002          51$:    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11283  052000  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11284  052002  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
11285  052006  000240                          NOP
11286
11287  052010  013737  001372  001142  52$:     MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11288  052016  042737  137777  001142          BIC     #^CSKI,$BDDAT   ;CLEAR DONT CARES
11289  052024  013737  052676  001140          MOV     300$,$GDDAT     ;GET EXPECTED SKI STATUS
11290  052032  042737  137777  001140          BIC     #^CSKI,$GDDAT   ;CLEAR DONT CARES
11291  052040  001417                          BEQ     53$             ;BRANCH IF 0 EXPECTED
11292
11293                                  ;REPORT ERROR IF SKI IS NOT SET (IAE WAS NOT DETECTED)
11294  052042  032737  040000  001142          BIT     #SKI,$BDDAT     ;WAS SKI DETECTED ??
11295  052050  001032                          BNE     54$             ;YES !!
11296  052052  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11297  052056  112776  000267  000000          MOVB    #267,@(SP)      ;WRITE ERROR NUMBER
11298  052064  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
11299  052070  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11300  052072  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11301  052076  000443                          BR      6$              ;GO TO NEXT ERROR CHECK
11302  052100                          53$:
```

```
11303
11304                                   ;REPORT ERROR IF SKI IS SET
11305  052100  032737  040000  001142            BIT     #SKI,$BDDAT     ;IS SKI SET ??
11306  052106  001413                            BEQ     54$             ;NO - SKI IS OK
11307  052110  062716  000004                    ADD     #4,(SP)         ;MOVE (SP) TO ERROR
11308  052114  112776  000054  000000            MOVB    #54,@(SP)       ;LOAD ERROR NUMBER
11309  052122  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11310  052126  004736                            JSR     PC,@(SP)+       ;REPORT SEEK ERROR
11311  052130  162716  000010                    SUB     #10,(SP)        ;RESTORE (SP)
11312  052134  000240                            NOP
11313
11314                                   ;REPORT ANY DEVICE CHECK
11315  052136  032737  000200  001372  54$:      BIT     #DVC,RMER2I     ;WAS THERE DVC DURING SEEK??
11316  052144  001420                            BEQ     6$              ;NO!!
11317  052146  005037  001140                    CLR     $GDDAT          ;EXPECTED STATUS
11318  052152  013737  001372  001142            MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11319  052160  062716  000004                    ADD     #4,(SP)
11320  052164  112776  000055  000000            MOVB    #55,@(SP)       ;ERROR  #55
11321  052172  162716  000002                    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11322  052176  004736                            JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11323  052200  162716  000010                    SUB     #10,(SP)        ;RESTORE SP
11324  052204  000240                            NOP
11325
11326                                   ;REPORT ANY "OPI" ERROR AS OPI WITH MOL  =0, OR  OPI
11327                                   ;BECAUSE ON CYLINDER LATCH DIDN'T RESET
11328  052206  032737  020000  001344  6$:       BIT     #OPI,RMER1I     ;"OPI" ERROR??
11329  052214  001427                            BEQ     8$              ;NO!!
11330  052216  005037  001140                    CLR     $GDDAT          ;EXPECTED STATUS
11331  052222  013737  001344  001142            MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11332  052230  062716  000004                    ADD     #4,(SP)         ;MOVE (SP) TO ERROR
11333  052234  112776  000052  000000            MOVB    #52,@(SP)       ;LOAD ERROR NUMBER
11334  052242  032737  010000  001342            BIT     #MOL,RMDSI      ;WAS MEDIUM ON LINE??
11335  052250  001403                            BEQ     7$              ;NO!!
11336  052252  112776  000053  000000            MOVB    #53,@(SP)       ;YES - CHANGE ERROR NUMBER
11337  052260  162716  000002          7$:       SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11338  052264  004736                            JSR     PC,@(SP)+       ;REPORT "OPI" ERROR
11339  052266  162716  000010                    SUB     #10,(SP)        ;RESTORE (SP)
11340  052272  000240                            NOP
11341
11342                                   ;SEE IF "PIP" IS 0, AND "ATA","MOL"AND"VV" =1
11343  052274  013746  001342          8$:       MOV     RMDSI,-(SP)
11344  052300  042716  047677                    BIC     #^C<ATA!PIP!MOL!VV>,(SP)
11345  052304  022726  110100                    CMP     #ATA!MOL!VV,(SP)+
11346  052310  001002                            BNE     9$              ;ERROR IN RMDS
11347  052312  000137  052646                    JMP     14$             ;RMDS IS OK
11348
11349                                   ;REPORT ERROR IF MOL = 0 AND OPI = 0
11350  052316  032737  010000  001342  9$:       BIT     #MOL,RMDSI      ;IS MOL RESET??
11351  052324  001030                            BNE     10$             ;NO - MOL IS SET
11352  052326  032737  020000  001344            BIT     #OPI,RMER1I     ;WAS OPI SET
11353  052334  001024                            BNE     10$             ;YES - DONT REPORT ERROR
11354  052336  013737  001342  001140            MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11355  052344  052737  010000  001140            BIS     #MOL,$GDDAT
11356  052352  013737  001342  001142            MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11357  052360  062716  000004                    ADD     #4,(SP)
11358  052364  112776  000062  000000            MOVB    #62,@(SP)
```

```
11359  052372  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11360  052376  004736                          JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
11361  052400  162716  000010                  SUB     #10,(SP)
11362  052404  000240                          NOP
11363
11364                                  ;REPORT AN ERROR IF "PIP" IS STIL SET AND SKI NOT SET
11365  052406  032737  020000  001342  10$:     BIT     #PIP,RMDSI       ;IS "PIP" STILL SET??
11366  052414  001430                          BEQ     11$              ;NO!!
11367  052416  032737  040000  001372          BIT     #SKI,RMER2I      ;WAS "SKI"SET??
11368  052424  001024                          BNE     11$              ;YES-DONT REPORT PIP
11369  052426  013737  001342  001140          MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
11370  052434  042737  020000  001142          BIC     #PIP,$BDDAT
11371  052442  013737  001342  001142          MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
11372  052450  062716  000004                  ADD     #4,(SP)          ;MOVE (SP) TO ERROR
11373  052454  112776  000056  000000          MOVB    #56,@(SP)        ;LOAD ERROR NUMBER
11374  052462  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11375  052466  004736                          JSR     PC,@(SP)+        ;REPORT "PIP" SET AFTER SEEK
11376  052470  162716  000010                  SUB     #10,(SP)                  ;RESTORE (SP)
11377  052474  000240                          NOP
11378
11379                                  ;REPORT AN ERROR IF "ATA" IS NOT SET
11380  052476  032737  100000  001342  11$:     BIT     #ATA,RMDSI       ;WAS "ATA" SET ??
11381  052504  001024                          BNE     13$              ;YES!!
11382  052506  013737  001342  001140          MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
11383  052514  052737  110600  001140          BIS     #ATA!MOL!DPR!DRY,$GDDAT
11384  052522  013737  001342  001142          MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
11385  052530  062716  000004                  ADD     #4,(SP)          ;MOVE (SP) TO ERROR
11386  052534  112776  000057  000000          MOVB    #57,@(SP)        ;LOAD ERROR NUMBER
11387  052542  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11388  052546  004736                          JSR     PC,@(SP)+        ;REPORT ATTENTION NOT SET DURING
11389                                                                   ;SEEK TEST
11390  052550  162716  000010                  SUB     #10,(SP)                  ;RESTORE (SP)
11391  052554  000240                          NOP
11392
11393                                  ;REPORT ERROR IF VOLUME VALID IS RESET AND IVC IS ZERO
11394  052556  032737  000100  001342  13$:     BIT     #VV,RMDSI        ;IS VV = 0 ??
11395  052564  001030                          BNE     14$              ;NO!!
11396  052566  032737  010000  001372          BIT     #IVC,RMER2I      ;IS IVC ALSO 0 ??
11397  052574  001024                          BNE     14$              ;NO - IVC IS SET
11398  052576  013737  001342  001140          MOV     RMDSI,$GDDAT     ;EXPECTED STATUS
11399  052604  052737  000100  001140          BIS     #VV,$GDDAT
11400  052612  013737  001342  001142          MOV     RMDSI,$BDDAT     ;RECEIVED STATUS
11401  052620  062716  000004                  ADD     #4,(SP)
11402  052624  112776  000064  000000          MOVB    #64,@(SP)        ;ERROR #64
11403  052632  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11404  052636  004736                          JSR     PC,@(SP)+
11405  052640  162716  000010                  SUB     #10,(SP)
11406  052644  000240                          NOP
11407  052646                          14$:
11408
11409                                  ;MODIFY THE RETURN ADDRESS IF AN ERROR WAS DETECTED
11410  052646  000240                          NOP
11411  052650  062716  000004                  ADD     #4,(SP)          ;MOVE (SP) TO ERROR CALL
11412  052654  105776  000000                  TSTB    @(SP)            ;WAS ERROR CALLED??
11413  052660  001403                          BEQ     15$              ;NO!!
11414  052662  062716  000004                  ADD     #4,(SP)          ;MOVE TO ERROR RETURN
```

```
11415  052666  000402                        BR      16$
11416
11417  052670  162716  000004        15$:    SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
11418  052674  000207                16$:    RTS     PC              ;RETURN
11419
11420  052676  000000                300$:   .WORD   0               ;ERROR FLAGS
11421
```

```
                                        .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

053016                          CLRSTS:

                                ;CLEAR USER'S ERROR CALL
053016  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR
053022  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
053026  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR
                                ;REPORT ERROR IF RMCS1 NOT INITIALIZED
053032  013737  001330  001142  4$:     MOV     RMCS1I,$BDDAT   ;VERIFY RMCS1
053040  042737  100000  001142          BIC     #SC,$BDDAT      ;IGNORE SPECIAL CONDITION
053046  012737  004200  001140          MOV     #DVA!RDY,$GDDAT ;EXPECT DVA & RDY
053054  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED, RECEIVED
053062  001413                          BEQ     5$              ;BRANCH IF EQUAL
053064  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
053070  112776  000126  000000          MOVB    #126,@(SP)      ;WITE ERROR NUMBER IN CALL
053076  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
053102  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
053104  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
053110  000240                          NOP
                                ;REPORT ERROR IF RMBA NOT RESET
053112  005037  001140  5$:     CLR     $GDDAT                  ;VERIFY RMBA IS ZERO
053116  013737  001334  001142          MOV     RMBAI,$BDDAT
053124  001413                          BEQ     7$              ;BRANCH IF ZERO
053126  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
053132  112776  000127  000000          MOVB    #127,@(SP)      ;WITE ERROR NUMBER IN CALL
053140  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
053144  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
053146  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
053152  000240                          NOP
                                ;REPORT ERROR IF RMCS2 NOT INITIALIZED
053154  013737  001340  001142  7$:     MOV     RMCS2I,$BDDAT   ;VERIFY RMCS2
053162  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
053164  005046                          CLR     -(SP)           ;EXPECT IR & UNIT NUMBER
053166  013701  001450                  MOV     TSTQUE,R1       ;R1 = ADDRESS OF TEST QUE
053172  111116                          MOVB    (R1),(SP)
053174  052716  000100                  BIS     #IR,(SP)
```

```
11512  053200  012637  001140                 MOV     (SP)+,$GDDAT
11513  053204  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
11514  053206  023737  001140  001142         CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11515  053214  001413                         BEQ     9$              ;BRANCH IF EQUAL
11516  053216  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11517  053222  112776  000130  000000         MOVB    #130,@(SP)      ;WITE ERROR NUMBER IN CALL
11518  053230  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11519  053234  004736                         JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11520  053236  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11521  053242  000240                         NOP
11522                          ;REPORT ERROR IF RMER1 NOT RESET-IGNORE UNS
11523  053244  005037  001140          9$:    CLR     $GDDAT          ;VERIFY RMER1
11524  053250  013737  001344  001142         MOV     RMER1I,$BDDAT
11525  053256  042737  040000  001142         BIC     #UNS,$BDDAT     ;IGNORE UNSAFE
11526  053264  001413                         BEQ     13$             ;BRANCH IF ZERO
11527  053266  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11528  053272  112776  000131  000000         MOVB    #131,@(SP)      ;WITE ERROR NUMBER IN CALL
11529  053300  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11530  053304  004736                         JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11531  053306  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11532  053312  000240                         NOP
11533                          ;REPORT ERROR IF RMMR1 NOT INITIALIZED-IGNORE WC,LS,LST
11534  053314  013737  001354  001142  13$:    MOV     RMMR1I,$BDDAT   ;VERIFY RMMR
11535  053322  042737  000046  001142         BIC     #WC!LS!LST,$BDDAT       ;IGNORE WORD CLOCK, SCT, TRK
11536  053330  012737  000010  001140         MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA BIT
11537  053336  023737  001140  001142         CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
11538  053344  001413                         BEQ     17$             ;BRANCH IF 0
11539  053346  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11540  053352  112776  000133  000000         MOVB    #133,@(SP)      ;WITE ERROR NUMBER IN CALL
11541  053360  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11542  053364  004736                         JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11543  053366  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11544  053372  000240                         NOP
11545                          ;REPORT AN ERROR IF RMEC2 IS NOT RESET
11546  053374  005037  001140          17$:   CLR     $GDDAT          ;EXPECT ZEROS
11547  053400  013737  001376  001142         MOV     RMEC2I,$BDDAT   ;VERIFY RMEC2=0
11548  053406  001413                         BEQ     19$
11549  053410  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11550  053414  112776  000135  000000         MOVB    #135,@(SP)      ;WITE ERROR NUMBER IN CALL
11551  053422  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11552  053426  004736                         JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11553  053430  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11554  053434  000240                         NOP
11555                          ;REPORT ERROR IF RMMR2 NOT INITIALIZED-IGNORE RQA,RQB
11556  053436  013737  001370  001142  19$:    MOV     RMMR2I,$BDDAT   ;VERIFY RMMR2
11557  053444  042737  140000  001142         BIC     #RQA!RQB,$BDDAT
11558  053452  012737  011777  001140         MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
11559  053460  023737  001140  001142         CMP     $GDDAT,$BDDAT
11560  053466  001413                         BEQ     21$
11561  053470  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11562  053474  112776  000136  000000         MOVB    #136,@(SP)      ;WITE ERROR NUMBER IN CALL
11563  053502  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11564  053506  004736                         JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11565  053510  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11566  053514  000240                         NOP
11567                          ;REPORT ERROR IF RMER2 NOT RESET-IGNORE SKI,DVC
```

```
11568  053516  005037  001140          21$:    CLR     $GDDAT          ;EXPECT ALL ZEROS
11569  053522  013737  001372  001142          MOV     RMER2I,$BDDAT   ;VERIFY RMER2
11570  053530  042737  040200  001142          BIC     #SKI!DVC,$BDDAT ;IGNORE DEVICE ERRORS
11571  053536  001413                          BEQ     21S$            ;BRANCH IF OTHER BITS 0
11572  053540  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11573  053544  112776  000174  000000          MOVB    #174,@(SP)      ;WITE ERROR NUMBER IN CALL
11574  053552  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11575  053556  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11576  053560  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11577  053564  000240                          NOP
11578                                  ;REPORT ERROR IF RMDS NOT INITIALIZED
11579  053566  013737  001342  001142  21S$:    MOV     RMDSI,$BDDAT    ;TEST DRIVE STATUS REGISTER
11580  053574  042737  177177  001142          BIC     #^C<DRY!DPR>,$BDDAT
11581  053602  012737  000600  001140          MOV     #DPR!DRY,$GDDAT ;EXPECTED DRIVE STATUS
11582  053610  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
11583  053616  001413                          BEQ     22$             ;BRANCH IF EQUAL
11584  053620  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11585  053624  112776  000134  000000          MOVB    #134,@(SP)      ;WRITE ERROR NUMBER
11586  053632  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11587  053636  004736                          JSR     PC,@(SP)+       ;REPORT ERROR TO USER
11588  053640  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11589  053644  000240                          NOP
11590  053646  062716  000004          22$:    ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11591  053652  105776  000000                  TSTB    @(SP)           ;WAS AN ERROE DETECTED??
11592  053656  001403                          BEQ     23$             ;NO!!
11593  053660  062716  000004                  ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
11594  053664  000402                          BR      24$
11595  053666  162716  000004          23$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11596  053672  000240                  24$:    NOP
11597  053674  000207                          RTS     PC
```

```
11598                                   .SBTTL  PACK ACKNOWLEDGE STATUS CHECK
11599
11600                                   ;THIS SUBROUTINE CHECKS THE RESULTS OF A PACK ACKNOWLEDGE
11601                                   ;COMMAND USING THE STATUS STORED IN THE GET BUFFER.  ERRORS ARE
11602                                   ;REPORTED TO THE USER VIA THE USER'S ERROR CALL.
11603
11604                                   ;CALL:
11605                                   ;(1)    JSR     PC,ACKSTS
11606                                   ;       BR      ???             RETURN HERE IF NO ERROR
11607                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
11608                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
11609                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
11610                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
11611
11612  053676                           ACKSTS:
11613
11614                                   ;CLEAR USER'S ERROR CALL
11615  053676  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11616  053702  105076  000000                  CLRB    @(SP)           ;CLEAR LOW ORDER BYTE
11617  053706  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK
11618
11619                                   ;REPORT AN ERROR  IF "VV" IS 0
11620  053712  032737  000100  001342          BIT     #VV,RMDSI       ;IS VOLUME VALID SET??
11621  053720  001024                          BNE     1$              ;YES!!
11622  053722  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11623  053730  052737  000100  001140          BIS     #VV,$GDDAT
11624  053736  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11625  053744  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11626  053750  112776  000155  000000          MOVB    #155,@(SP)      ;WRITE NUMBER IN ERROR CALL
11627  053756  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11628  053762  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR
11629  053764  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11630  053770  000240                          NOP
11631  053772                           1$:
11632
11633                                   ;REPORT AN ERROR IF "MOL" IS 0
11634  053772  032737  010000  001342          BIT     #MOL,RMDSI      ;IS MOL SET??
11635  054000  001024                          BNE     2$              ;YES!!
11636  054002  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11637  054010  052737  010000  001140          BIS     #MOL,$GDDAT
11638  054016  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11639  054024  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11640  054030  112776  000041  000000          MOVB    #41,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11641  054036  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11642  054042  004736                          JSR     PC,@(SP)+       ;REPORT TH ERROR
11643  054044  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO BRANCH
11644  054050  000240                          NOP
11645  054052                           2$:
11646
11647                                   ;SEE IF "UNS","OPI","RMR","ILR", OR "ILF" IS SET
11648  054052  032737  060007  001344          BIT     #UNS!OPI!RMR!ILR!ILF,RMER1I
11649  054060  001570                          BEQ     7$
11650
11651                                   ;REPORT AN ERROR IF "UNS" IS SET
11652  054062  032737  040000  001344          BIT     #UNS,RMER1I     ;WAS UNS SET??
11653  054070  001424                          BEQ     3$              ;NO!!
```

```
11654  054072  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11655  054100  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11656  054106  042737  040000  001140          BIC     #UNS,$GDDAT
11657  054114  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11658  054120  112776  000042  000000          MOVB    #42,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11659  054126  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11660  054132  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11661  054134  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11662  054140  000240                          NOP
11663  054142                          3$:
11664
11665                                  ;REPORT ANY OPI ERROR
11666  054142  032737  020000  001344          BIT     #OPI,RMER1I     ;WAS OPI SET ??
11667  054150  001424                          BEQ     4$              ;NO!!
11668  054152  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11669  054160  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11670  054166  042737  020000  001140          BIC     #OPI,$GDDAT
11671  054174  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11672  054200  112776  000043  000000          MOVB    #43,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11673  054206  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11674  054212  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11675  054214  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11676  054220  000240                          NOP
11677  054222                          4$:
11678
11679                                  ;REPORT ANY RMR ERROR
11680  054222  032737  000004  001344          BIT     #RMR,RMER1I     ;WAS RMR SET??
11681  054230  001424                          BEQ     5$              ;NO!!
11682  054232  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11683  054240  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11684  054246  042737  000004  001140          BIC     #RMR,$GDDAT
11685  054254  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11686  054260  112776  000044  000000          MOVB    #44,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11687  054266  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11688  054272  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11689  054274  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11690  054300  000240                          NOP
11691  054302                          5$:
11692
11693                                  ;REPORT ANY     ILR ERROR
11694  054302  032737  000002  001344          BIT     #ILR,RMER1I     ;WAS ILR SET??
11695  054310  001424                          BEQ     6$              ;NO!!
11696  054312  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11697  054320  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11698  054326  042737  000002  001140          BIC     #ILR,$GDDAT
11699  054334  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11700  054340  112776  000045  000000          MOVB    #45,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11701  054346  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11702  054352  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11703  054354  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11704  054360  000240                          NOP
11705  054362                          6$:
11706
11707                                  ;REPORT ANY     ILF ERROR
11708  054362  032737  000001  001344          BIT     #ILF,RMER1I     ;WAS ILF SET??
11709  054370  001424                          BEQ     7$              ;NO!!
```

```
11710  054372  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11711  054400  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11712  054406  042737  000001  001140          BIC     #ILF,$GDDAT
11713  054414  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11714  054420  112776  000046  000000          MOVB    #46,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
11715  054426  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11716  054432  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
11717  054434  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11718  054440  000240                          NOP
11719  054442                          7$:
11720
11721                                  ;AUGMENT RETURN ADDRESS IF ERROR WAS FOUND
11722  054442  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11723  054446  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
11724  054452  001403                          BEQ     8$              ;NO!!
11725  054454  062716  000004                  ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
11726  054460  000402                          BR      9$
11727  054462  162716  000004          8$:     SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11728  054466  000240                  9$:     NOP
11729  054470  000207                          RTS     PC
11730
```

```
11731                                   .SBTTL  RECALIBRATE STATUS CHECK SUBROUTINE
11732
11733                                   ;THIS SUBROUTINE CHECKS THE RESULTS OF A RECALIBRATE OPERATION
11734                                   ;USING THE STATUS STORED IN THE GET BUFFER.
11735
11736                                   ;CALL:
11737                                   ;
11738                                   ;(1)    JSR     PC,RCLSTS       ;CALL SUBROUTINE
11739                                   ;       BR      ???             ;RETURN HERE IF NO ERROR
11740                                   ;       NOP                     ;RETURN HERE TO REPORT AN ERROR
11741                                   ;       ERROR                   ;ERROR NUMBER DEFINED BY SUB
11742                                   ;       JSR     PC,@(SP)+       ;GO BACK TO SUB FOR MORE ERROR CHECKS
11743                                   ;       ???                     ;RETURN HERE IF NO MORE ERRORS
11744
11745   054472                          RCLSTS:
11746
11747                                   ;CLEAR USER'S ERROR NUMBER
11748   054472  062716  000004                  ADD     #4,(SP)
11749   054476  105076  000000                  CLRB    @(SP)           ;CLEAR USER'S ERROR CALL
11750   054502  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO BRANCH
11751
11752
11753                                   ;SEE IF "PAR" OR "ILF" OR "OPI" OR "IAE" IS SET
11754   054506  032737  022011  001344          BIT     #OPI!PAR!ILF!IAE,RMER1I
11755   054514  001553                  BEQ     4$              ;NONE ARE SET - GO TO NEXT CHECK
11756
11757                                   ;REPORT ANY MASSBUS CONTROL BUS PARITY ERROR, I.E.,
11758                                   ;"PAR" = 1 AND "DPE" = 0
11759   054516  032737  000010  001344          BIT     #PAR,RMER1I     ;WAS "PAR" SET??
11760   054524  001430                  BEQ     1$              ;NO!!
11761   054526  032737  000010  001372          BIT     #DPE,RMER2I     ;WAS "DPE" SET??
11762   054534  001024                  BNE     1$              ;YES - NOT A REGISTER ERROR
11763   054536  013737  001344  001140          MOV RMER1I,$GDDAT       ;EXPECTED STATUS
11764   054544  042737  000010  001140          BIC     #PAR,$GDDAT
11765   054552  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11766   054560  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11767   054564  112776  000050  000000          MOVB    #50,@(SP)       ;WRITE ERROR NUMBER IN CALL
11768   054572  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11769   054576  004736                  JSR     PC,@(SP)+       ;GO REPORT ERROR
11770   054600  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11771   054604  000240                  NOP
11772   054606                          1$:
11773
11774                                   ;REPORT ANY "ILF" ERROR
11775   054606  032737  000001  001344          BIT     #ILF,RMER1I     ;WAS "ILF" SET??
11776   054614  001424                  BEQ     2$              ;NO!!
11777   054616  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11778   054624  042737  000001  001140          BIC     #ILF,$GDDAT
11779   054632  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11780   054640  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11781   054644  112776  000071  000000          MOVB    #71,@(SP)       ;WRITE ERROR NUMBER IN CALL
11782   054652  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11783   054656  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11784   054660  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11785   054664  000240                  NOP
11786   054666                          2$:
```

```
11787
11788                                   ;REPORT ANY "OPI" ERROR AS
11789                                   ;       . OPI DUE TO "MOL" = 0
11790                                   ;       . OPI BECAUSE ON CYLINDER LATCH DIDN'T RESET
11791  054666  032737  020000  001344           BIT     #OPI,RMER1I     ;WAS OPI SET??
11792  054674  001433                           BEQ     31$             ;NO!!
11793  054676  013737  001344  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11794  054704  042737  020000  001140           BIC     #OPI,$GDDAT
11795  054712  013737  001344  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11796  054720  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11797  054724  112776  000072  000000           MOVB    #72,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11798  054732  032737  010000  001342           BIT     #MOL,RMDSI      ;WAS "MOL" = 0??
11799  054740  001403                           BEQ     3$              ;YES!!
11800  054742  112776  000073  000000           MOVB    #73,@(SP)       ;NO - CHANGE ERROR NUMBER
11801  054750  162716  000002           3$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11802  054754  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11803  054756  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11804  054762  000240                           NOP
11805  054764                           31$:
11806
11807                                   ;REPORT AN ERROR IF "IAE" IS SET
11808  054764  032737  002000  001344           BIT     #IAE,RMER1I     ;IS "IAE" SET??
11809  054772  001424                           BEQ     4$              ;NO!!
11810  054774  013737  001344  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11811  055002  042737  002000  001140           BIC     #IAE,$GDDAT
11812  055010  013737  001344  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11813  055016  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
11814  055022  112776  000070  000000           MOVB    #70,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11815  055030  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11816  055034  004736                           JSR     PC,@(SP)+       ;REPORT ERROR
11817  055036  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
11818  055042  000240                           NOP
11819  055044                           4$:
11820
11821                                   ;SEE IF "SKI" OR "IVC" OR "DVC" IS SET
11822  055044  032737  050200  001372           BIT     #SKI!IVC!DVC,RMER2I
11823  055052  001517                           BEQ     8$              ;NONE OF THE BITS ARE SET
11824
11825
11826                                   ;REPORT ANY "IVC" ERROR AS
11827                                   ;       .IVC WITH VV = 0
11828                                   ;       .ERRONEOUS IVC ERROR
11829  055054  032737  010000  001372           BIT     #IVC,RMER2I     ;WAS IVC SET??
11830  055062  001433                           BEQ     6$              ;NO!!
11831  055064  013737  001372  001140           MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11832  055072  042737  010000  001140           BIC     #IVC,$GDDAT
11833  055100  013737  001372  001142           MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11834  055106  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11835  055112  112776  000074  000000           MOVB    #74,@(SP)       ;;WRITE ERROR NUMBER IN CALL
11836  055120  032737  000100  001342           BIT     #VV,RMDSI       ;WAS VV = 0??
11837  055126  001403                           BEQ     5$              ;YES!!
11838  055130  112776  000075  000000           MOVB    #75,@(SP)       ;NO - CHANGE ERROR NUMBER
11839  055136  162716  000002           5$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11840  055142  004736                           JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11841  055144  162716  000010                   SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
11842  055150  000240                           NOP
```

```
11843   055152                                  6$:
11844
11845                                           ;REPORT ANY "SKI" ERROR
11846   055154  032737  040000  001372                  BIT     #SKI,RMER2I     ;WAS SKI SET??
11847   055162  001424                                  BEQ     7$              ;NO!!
11848   055164  013737  001372  001140                  MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11849   055172  042737  040000  001140                  BIC     #SKI,$GDDAT
11850   055176  013737  001372  001142                  MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11851   055204  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11852   055210  112776  000076  000000                  MOVB    #76,@(SP)       ;WRITE ERROR NUMBER
11853   055216  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11854   055222  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11855   055224  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO BRANCH
11856   055230  000240                                  NOP
11857   055232                                  7$:
11858
11859                                           ;REPORT ANY "DVC" ERROR
11860   055232  032737  000200  001372                  BIT     #DVC,RMER2I     ;WAS "DVC" SET??
11861   055240  001424                                  BEQ     8$              ;NO!!
11862   055242  013737  001372  001140                  MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11863   055250  042737  000200  001140                  BIC     #DVC,$GDDAT
11864   055256  013737  001372  001142                  MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11865   055264  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11866   055270  112776  000077  000000                  MOVB    #77,@(SP)       ;WRITE ERROR NUMBER
11867   055276  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11868   055302  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11869   055304  162716  000010                          SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11870   055310  000240                                  NOP
11871   055312                                  8$:
11872
11873                                           ;SEE IF "PIP" AND "OM" ARE 0, AND "ATA","MOL" AND "VV" ARE 1
11874   055312  013746  001342                          MOV     RMDSI,-(SP)     ;PUT RMDS ON STACK
11875   055316  042716  047676                          BIC     #^C<PIP!MOL!VV!OM!ATA>,(SP)
11876   055322  022726  110100                          CMP     #ATA!MOL!VV,(SP)+
11877   055326  001002                                  BNE     85$
11878   055330  000137  055744                          JMP     13$
11879   055334                                  85$:
11880
11881                                           ;REPORT AN ERROR IF MOL = 0 AND OPI = 0, I.E., MEDIUM WENT OFF
11882                                           ;LINE AFTER RECALIBRATE WAS INITIATED
11883   055334  032737  010000  001342                  BIT     #MOL,RMDSI      ;DID MOL DROP??
11884   055342  001030                                  BNE     9$              ;NO!!
11885   055344  032737  020000  001344                  BIT     #OPI,RMER1I     ;WAS OPI ERROR REPORTED??
11886   055352  001024                                  BNE     9$              ;YES - DON'T REPORT MOL=0
11887   055354  013737  001342  001140                  MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11888   055362  052737  010000  001140                  BIS     #MOL,$GDDAT
11889   055370  013737  001342  001142                  MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11890   055376  062716  000004                          ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11891   055402  112776  000100  000000                  MOVB    #100,@(SP)      ;WRITE ERROR NUMBER
11892   055410  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11893   055414  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11894   055416  162716  000010                          SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
11895   055422  000240                                  NOP
11896   055424                                  9$:
11897
11898                                           ;REPORT AN ERROR IF "VV" = 0 AND "IVC" = 0
```

```
11899  055424  032737  000100  001342          BIT     #VV,RMDSI       ;DID "VV" DROP??
11900  055432  001030                          BNE     10$             ;NO!!
11901  055434  032737  010000  001372          BIT     #IVC,RMER2I     ;WAS THERE A IVC ERROR??
11902  055442  001024                          BNE     10$             ;YES - DONT REPORT VV=0
11903  055444  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11904  055452  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11905  055460  052737  000100  001140          BIS     #VV,$GDDAT
11906  055466  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11907  055472  112776  000101  000000          MOVB    #101,@(SP)      ;WRITE ERROR NUMBER IN CALL
11908  055500  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11909  055504  004736                          JSR     PC,@(SP)+
11910  055506  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
11911  055512  000240                          NOP
11912  055514                          10$:
11913
11914                                  ;REPORT AN ERROR IF ATA IS NOT SET
11915  055514  032737  100000  001342          BIT     #ATA,RMDSI      ;WAS ATA SET DURING RECALIBRATE??
11916  055522  001024                          BNE     11$             ;YES!!
11917  055524  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11918  055532  052737  100000  001140          BIS     #ATA,$GDDAT
11919  055540  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11920  055546  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11921  055552  112776  000102  000000          MOVB    #102,@(SP)      ;WRITE ERROR NUMBER IN CALL
11922  055560  162716  000002                  SUB     #2,(SP)
11923  055564  004736                          JSR     PC,@(SP)+
11924  055566  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11925  055572  000240                          NOP
11926
11927  055574                          11$:
11928
11929                                  ;REPORT AN ERROR IF "OM" IS NOT ZERO BECAUSE RECALIBRATE SHOULD
11930                                  ;ALWAYS CLEAR OFFSET MODE
11931  055574  032737  000001  001342          BIT     #OM,RMDSI       ;WAS "OM" RESET??
11932  055602  001424                          BEQ     12$             ;YES!!
11933  055604  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11934  055612  042737  000001  001140          BIC     #OM,$GDDAT
11935  055620  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11936  055626  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11937  055632  112776  000103  000000          MOVB    #103,@(SP)      ;WRITE ERROR NUMBER
11938  055640  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11939  055644  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11940  055646  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
11941  055652  000240                          NOP
11942  055654                          12$:
11943
11944                                  ;REPORT AN ERROR IF "PIP" IS STIL ON, I.E., DRIVE NOT ON
11945                                  ;CYLINDER
11946  055654  032737  020000  001342          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
11947  055662  001430                          BEQ     13$             ;NO!!
11948  055664  032737  040000  001372          BIT     #SKI,RMER2I     ;WAS "SKI" DETECTED??
11949  055672  001024                          BNE     13$             ;YES-DONT REPORT "PIP"
11950  055674  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11951  055702  042737  020000  001140          BIC     #PIP,$GDDAT
11952  055710  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11953  055716  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11954  055722  112776  000104  000000          MOVB    #104,@(SP)      ;WRITE ERROR NUMBER
```

```
11955  055730  162716  000002          SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11956  055734  004736                  JSR     PC,@(SP)+
11957  055736  162716  000010          SUB     #10,(SP)         ;MOVE SP BACK TO USER'S BRANCH
11958  055742  000240                  NOP
11959  055744                      13$:
11960
11961                              ;SEE IF "ILR" OR "RMR" OR "UNS" IS SET
11962  055744  032737  040006  001344  BIT     #ILR!RMR!UNS,RMER1I
11963  055752  001514                  BEQ     16$
11964
11965                              ;REPORT AN ERROR IF "ILR" IS SET
11966  055754  032737  000002  001344  BIT     #ILR,RMER1I      ;WAS ILR SET DURING RECALIBRATE??
11967  055762  001424                  BEQ     14$              ;NO!!
11968  055764  013737  001344  001140  MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
11969  055772  042737  000002  001140  BIC     #ILR,$GDDAT
11970  056000  013737  001344  001142  MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
11971  056006  062716  000004          ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11972  056012  112776  000105  000000  MOVB    #105,@(SP)       ;WRITE ERROR NUMBER IN CALL
11973  056020  162716  000002          SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11974  056024  004736                  JSR     PC,@(SP)+
11975  056026  162716  000010          SUB     #10,(SP)         ;MOVE SP TO USER'S BRANCH
11976  056032  000240                  NOP
11977  056034                      14$:
11978
11979                              ;REPORT AN ERROR IF "RMR" IS SET
11980  056034  032737  000004  001344  BIT     #RMR,RMER1I      ;WAS RMR SET??
11981  056042  001424                  BEQ     15$              ;NO!!
11982  056044  013737  001344  001140  MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
11983  056052  042737  000004  001140  BIC     #RMR,$GDDAT
11984  056060  013737  001344  001142  MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
11985  056066  062716  000004          ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
11986  056072  112776  000106  000000  MOVB    #106,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11987  056100  162716  000002          SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
11988  056104  004736                  JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
11989  056106  162716  000010          SUB     #10,(SP)         ;MOVE SP TO USER'S BRANCH
11990  056112  000240                  NOP
11991  056114                      15$:
11992
11993                              ;REPORT AN ERROR IF "UNS" IS SET AND "DVC" IS 0
11994  056114  032737  040000  001344  BIT     #UNS,RMER1I      ;WAS UNSAFE ON??
11995  056122  001430                  BEQ     16$              ;NO!!
11996  056124  032737  000200  001372  BIT     #DVC,RMER2I      ;WAS THERE A DEVICE CHECK??
11997  056132  001024                  BNE     16$              ;YES - DON'T REPORT UNSAFE
11998  056134  013737  001344  001140  MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
11999  056142  042737  040000  001140  BIC     #UNS,$GDDAT
12000  056150  013737  001344  001142  MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
12001  056156  062716  000004          ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
12002  056162  112776  000107  000000  MOVB    #107,@(SP)       ;WRITE ERROR NUMBER
12003  056170  162716  000002          SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
12004  056174  004736                  JSR     PC,@(SP)+        ;REPORT ERROR VIA USER
12005  056176  162716  000010          SUB     #10,(SP)         ;MOVE SP BACK TO USER'S BRANCH
12006  056202  000240                  NOP
12007  056204                      16$:
12008
12009                              ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
12010  056204  062716  000004          ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
```

```
12011  056210  105776  000000                TSTB    @(SP)              ;WAS AN ERROR REPORTED??
12012  056214  001403                        BEQ     17$                ;NO!!
12013  056216  062716  000004                ADD     #4,(SP)            ;YES - AUGMENT SP RETURN
12014  056222  000402                        BR      18$
12015  056224  162716  000004          17$:  SUB     #4,(SP) ;NO ERROR - RETURN SP TO BRANCH
12016  056230  000240                  18$:  NOP
12017  056232  000207                        RTS     PC                 ;STATUS CECK IS COMPLETE
12018
```

```
12019                                   .SBTTL  DRIVE CLEAR STATUS CHECK SUBROUTINE
12020                                   ;       BR      ???             RETURN HERE IF NO ERROR
12021                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
12022                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
12023                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
12024                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
12025
12026   056234                          DRVSTS:
12027
12028                                   ;CLEAR USER'S ERROR CALL
12029   056234  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12030   056240  105076  000000                 CLRB    @(SP)           ;CLEAR ERROR CALL
12031   056244  162716  000004                 SUB     #4,(SP)         ;MOVE SP TO USER'S BRANCH
12032                                   ;REPORT ERROR IF RMCS1 NOT INITIALIZED
12033   056250  013737  001330  001142 4$:     MOV     RMCS1I,$BDDAT   ;CHECK RMCS1
12034   056256  042737  173700  001142         BIC     #^C<DVA!FNCMSK>,$BDDAT  ;CLEAR DONT CARES
12035   056264  012737  004010  001140         MOV     #DVA!DRVCLR,$GDDAT      ;EXPECT DVA
12036   056272  023737  001140  001142         CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
12037   056300  001443                         BEQ     6$              ;BRANCH IF EQUAL
12038   056302  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12039   056306  112776  000141  000000         MOVB    #141,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12040   056314  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12041   056320  004736                         JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12042   056322  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12043   056326  000240                         NOP
12044                                   ;REPORT ERROR IF RMDS NOT INITIALIZED
12045   056330  013737  001342  001142 5$:     MOV     RMDSI,$BDDAT    ;CHECK RMDS
12046   056336  042737  021101  001142         BIC     #PGM!OM!VV!PIP,$BDDAT   ;CLEAR DONT CARES
12047   056344  012737  010600  001140         MOV     #MOL!DPR!DRY,$GDDAT     ;EXPECT DRY & DPR
12048   056352  023737  001140  001142         CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
12049   056360  001413                         BEQ     6$              ;BRANCH IF EQUAL
12050   056362  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12051   056366  112776  000142  000000         MOVB    #142,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12052   056374  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12053   056400  004736                         JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12054   056402  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12055   056406  000240                         NOP
12056                                   ;REPORT ERROR IF RMER1 NOT INITIALIZED
12057   056410  005037  001140         6$:     CLR     $GDDAT          ;EXPECT 0'S
12058   056414  013737  001344  001142         MOV     RMER1I,$BDDAT   ;CHECK RMER1
12059   056422  001413                         BEQ     8$              ;BRANCH IF EQUAL
12060   056424  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12061   056430  112776  000143  000000         MOVB    #143,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12062   056436  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12063   056442  004736                         JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12064   056444  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12065   056450  000240                         NOP
12066                                   ;REPORT ERROR IF ATA NOT INITIALIZED
12067   056452  013737  001346  001142 8$:     MOV     RMASI,$BDDAT    ;CHECK ATTENTION BIT
12068   056460  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12069   056462  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12070   056464  013701  001450                 MOV     TSTQUE,R1
12071   056470  116102  000001                 MOVB    1(R1),R2
12072   056474  042702  177400                 BIC     #^CATNMSK,R2
12073   056500  005102                         COM     R2
12074   056502  040237  001142                 BIC     R2,$BDDAT
```

```
12075  056506  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
12076  056510  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
12077  056512  005737  001142                TST     $BDDAT          ;IS ATTENTION CLEARED??
12078  056516  001413                        BEQ     9$              ;BRANCH IF ATTENTION CLEARED
12079  056520  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12080  056524  112776  000144  000000        MOVB    #144,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12081  056532  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12082  056536  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12083  056540  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12084  056544  000240                        NOP
12085                          ;REPORT ERROR IF RMMR1 NOT INITIALIZED
12086  056546  013737  001354  001142  9$:   MOV     RMMR1I,$BDDAT   ;CHECK RMMR
12087  056554  042737  000046  001142        BIC     #WC!LS!LST,$BDDAT       ;CLEAR DONT CARES
12088  056562  012737  000010  001140        MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA ON
12089  056570  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
12090  056576  001413                        BEQ     11$             ;BRANCH IF ZERO
12091  056600  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12092  056604  112776  000145  000000        MOVB    #145,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12093  056612  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12094  056616  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12095  056620  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12096  056624  000240                        NOP
12097                          ;REPORT ERROR IF RMMR2 NOT INITIALIZED
12098  056626  013737  001370  001142  11$:   MOV     RMMR2I,$BDDAT   ;CHECK RMMR2
12099  056634  042737  140000  001142        BIC     #RQA!RQB,$BDDAT ;CLEAR REQA, REQB
12100  056642  012737  011777  001140        MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
12101  056650  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
12102  056656  001413                        BEQ     15$             ;BRANCH IF EQUAL
12103  056660  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12104  056664  112776  000146  000000        MOVB    #146,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12105  056672  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12106  056676  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12107  056700  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12108  056704  000240                        NOP
12109  056706  005037  001140        15$:    CLR     $GDDAT          ;EXPECT ZEROS
12110                          ;REPORT ERROR IF RMEC2 NOT RESET
12111  056712  013737  001376  001142        MOV     RMEC2I,$BDDAT   ;CHECK RMEC2
12112  056720  001413                        BEQ     17$             ;BRANCH IF 0
12113  056722  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12114  056726  112776  000150  000000        MOVB    #150,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12115  056734  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12116  056740  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12117  056742  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12118  056746  000240                        NOP
12119                          ;REPORT ERROR IF RMER2 NOT RESET
12120  056750  013737  001372  001142  17$:   MOV     RMER2I,$BDDAT   ;CHECK RMER2
12121  056756  001413                        BEQ     18$             ;BRANCH IF NO ERROR
12122  056760  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12123  056764  112776  000147  000000        MOVB    #147,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12124  056772  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12125  056776  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
12126  057000  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12127  057004  000240                        NOP
12128  057006                         18$:
12129
12130  057006                         19$:
```

```
12131
12132                                   ;AUGMENT RETURN ADDRESS IF ANY ERROR WAS FOUND
12133   057006  062716  000004                ADD     #4,(SP)           ;MOVE SP TO ERROR CALL
12134   057012  105776  000000                TSTB    @(SP)             ;WAS AN ERROR DETECTED??
12135   057016  001403                        BEQ     21$               ;NO!!
12136   057020  062716  000004                ADD     #4,(SP)           ;YES - MOVE SP TO ERROR RETURN
12137   057024  000402                        BR      23$
12138   057026  162716  000004        21$:    SUB     #4,(SP) ;MOVE SP BACK TO NO ERROR RETURN
12139   057032  000240                23$:    NOP
12140   057034  000207                        RTS     PC                ;RETURN TO USER
```

```
12141                                   .SBTTL  DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE
12142
12143                                   ;THIS SUBROUTINE VERIFIES THE RESULTS OF ALL DATA TRANSFER COMMANDS
12144                                   ;USING STATUS STORED IN THE GET BUFFER AND TEST PARAMETERS
12145                                   ;STORED IN THE PUT BUFFER.  ERRORS ARE REPORTED BY WRITING
12146                                   ;THE ERROR NUMBER IN THE USERS ERROR CALL.
12147
12148                                   ;USER'S SUBROUTINE CALL:
12149                                   ;(1)    JSR     PC,DTASTS
12150                                   ;(2)    BR      ??              RETURN HERE IF NO DATA ERRORS
12151                                   ;(3)    NOP                     RETURN HERE TO REPORT AN ERROR
12152                                   ;(4)    ERROR                   SUB WRITES ERROR NUMBER
12153                                   ;(5)    JSR     PC,@(SP)+       USER RETURNS FOR MORE CHECKS
12154                                   ;(6)    ??                      SUB RETURNS HERE AFTER ALL
12155                                   ;                               ERRORS ARE REPORTED
12156
12157   057036                          DTASTS:
12158
12159                                   ;CLEAR USER'S ERROR CALL AND ERROR FLAGS
12160   057036  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12161   057042  105076  000000                CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF TRAP
12162   057046  162716  000004                SUB     #4,(SP)         ;RESTORE SP TO NO ERROR
12163   057052  005037  062432                CLR     500$            ;CLEAR ERROR FLAGS
12164
12165                                   ;REPORT ANY CONTROL BUS PARITY ERROR WHILE READING REMOTE REGISTERS, I.E., MCPE = 1
12166   057056  032737  020000  001330        BIT     #MCPE,RMCS1I    ;WAS THERE A PARITY ERROR??
12167   057064  001422                        BEQ     10$             ;NO!!
12168   057066  013737  001330  001140        MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
12169   057074  042737  020000  001140        BIC     #MCPE,$GDDAT
12170   057102  013737  001330  001142        MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
12171   057110  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12172   057114  112776  000013  000000        MOVB    #13,@(SP)       ;WRITE ERROR NUMBER
12173   057122  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12174   057126  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12175   057130  000466                        BR      30$
12176   057132                          10$:
12177
12178                                   ;REPORT ANY CONTROL BUS PARITY ERROR WHILE WRING REMOTE REGISTERS, I.E.,
12179                                   ;PAR=1  AND DPE = 0
12180   057132  032737  000010  001344        BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
12181   057140  001435                        BEQ     20$             ;NO!!
12182   057142  032737  000010  001372        BIT     #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
12183   057150  001031                        BNE     20$             ;NO!!
12184   057152  013737  001344  001140        MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12185   057160  042737  000010  001140        BIC     #PAR,$GDDAT
12186   057166  013737  001344  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12187   057174  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12188   057200  112776  000050  000000        MOVB    #50,@(SP)       ;WRITE ERROR NUMBER
12189   057206  032737  001000  001330        BIT     #MXF,RMCS1I     ;DID MXF GET SET??
12190   057214  001003                        BNE     15$             ;YES!!
12191   057216  112776  000274  000000        MOVB    #274,@(SP)      ;NO - CHANGE ERROR NUMBER
12192   057224  162716  000002          15$:  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12193   057230  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12194   057232  000425                        BR      30$
12195
12196   057234                          20$:
```

```
12197
12198                                   ;LOOK FOR ANY ERRORS WHICH MAY HAVE OCCURRED DURING COMMAND INITIATION OR
12199                                   ;MECHANICAL POSITIONING
12200
12201                                   ;FIRST TEST MXF WHICH WOULD INDICATE COMPOSITE ERROR SET WHEN FUNCTION
12202                                   ;CODE AND GO BIT WERE LOADED
12203  057234  032737  001000  001340           BIT     #MXF,RMCS2I     ;WAS "MISSED TRANSFER" SET??
12204  057242  001425                         BEQ     40$             ;NO!!
12205  057244  013737  001340  001140         MOV     RMCS2I,$GDDAT   ;EXPECTED STATUS
12206  057252  042737  001000  001140         BIC     #MXF,$GDDAT
12207  057260  013737  001340  001142         MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
12208  057266  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12209  057272  112776  000275  000000         MOVB    #275,@(SP)      ;WRITE ERROR NUMBER
12210  057300  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12211  057304  004736                         JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12212  057306                          30$:
12213
12214                                   ;RESTORE SP TO NO ERROR RETURN AND BYPASS FURHTER STATUS CHECKING
12215  057306  162716  000010                 SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12216  057312  000137  062404                 JMP     380$            ;SKIP TO END OF SUB
12217
12218  057316                          40$:
12219
12220                                   ;REPORT AN ERROR IF "OPI" ERROR OCCURRED DUE TO "MOL" = 0, OR IF "OPI"
12221                                   ;AND "MOL" ARE SET, BUT "VV" IS RESET, INDICATING AN INTERMITTENT
12222                                   ;"MOL"
12223  057316  032737  020000  001344           BIT     #OPI,RMER1I     ;IS "OPI" SET??
12224  057324  001447                         BEQ     60$             ;NO!!
12225  057326  013737  001344  001140         MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12226  057334  042737  020000  001140         BIC     #OPI,$GDDAT
12227  057342  013737  001344  001142         MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12228  057350  032737  010000  001342         BIT     #MOL,RMDSI      ;WAS MEDIUM OFF LINE??
12229  057356  001404                         BEQ     45$             ;YES!!
12230  057360  032737  000100  001342         BIT     #VV,RMDSI       ;WAS "MOL" INTERMITTENT??
12231  057366  001013                         BNE     50$             ;NO!!
12232  057370  062716  000004          45$:   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12233  057374  112776  000276  000000         MOVB    #276,@(SP)      ;WRITE ERROR NUMBER IN CALL
12234  057402  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12235  057406  004736                         JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12236  057410  162716  000010                 SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12237  057414  000413                         BR      60$
12238  057416                          50$:
12239
12240                                   ;REPORT "OPI" ERROR, WHICH IS DUE TO "ON CYLINDER" NOT DROPPING OR
12241                                   ;"RUN" TIMEOUT (20 MS) OR SEARCH TIMEOUT (50 MS)
12242  057416  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12243  057422  112776  000277  000000         MOVB    #277,@(SP)      ;WRITE ERROR NUMBER IN CALL
12244  057430  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12245  057434  004736                         JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12246  057436  162716  000010                 SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12247  057442  000240                         NOP
12248  057444                          60$:
12249
12250                                   ;LOOK FOR "IVC" ERROR DURING COMMAND INITIATION
12251  057444  032737  010000  001372         BIT     #IVC,RMER2I     ;WAS THERE AN "IVC" ERROR??
12252  057452  001432                         BEQ     70$             ;NO!!
```

```
12253                                   ;REPORT "IVC" ERROR DUE TO "VV" = 0, OR REPORT ERRONEOUS "IVC" ERROR
12254  057454  013737  001372  001140             MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12255  057462  042737  010000  001140             BIC     #IVC,$GDDAT
12256  057470  013737  001372  001142             MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12257  057476  062716  000004                     ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12258  057502  112776  000300  000000             MOVB    #300,@(SP)      ;WRITE ERROR NUMBER IN CALL
12259  057510  032737  000100  001342             BIT     #VV,RMDSI       ;WAS VOLUME VALID??
12260  057516  001403                             BEQ     65$             ;NO!!
12261  057520  112776  000301  000000             MOVB    #301,@(SP)      ;CHANGE ERROR NUMBER
12262  057526  162716  000002           65$:      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12263  057532  004736                             JSR     PC,@(SP)+       ;REPORT "IVC" ERROR AND RETURN
12264  057534  162716  000010                     SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12265  057540                           70$:
12266
12267                                   ;SEE IF "ILF" OR "RMR" IS SET
12268  057540  032737  000007  001344             BIT     #ILR!ILF!RMR, RMER1I
12269  057546  001510                             BEQ     100$            ;NO ERRORS DETECTED
12270                                   ;REPORT AN ERROR IF "ILR" IS SET
12271  057550  032737  000002  001344             BIT     #ILR,RMER1I     ;WAS "ILR" DETECTED??
12272  057556  001424                             BEQ     80$             ;NO!!
12273  057560  013737  001344  001140             MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12274  057566  042737  000002  001140             BIC     #ILR,$GDDAT
12275  057574  013737  001344  001142             MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12276  057602  062716  000004                     ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12277  057606  112776  000302  000000             MOVB    #302,@(SP)      ;WRITE ERROR NUMBER IN CALL
12278  057614  162716  000002                     SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12279  057620  004736                             JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12280  057622  162716  000010                     SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12281  057626  000240                             NOP
12282  057630                           80$:
12283
12284                                   ;REPORT AN ERROR IF "ILF" IS SET
12285  057630  032737  000001  001344             BIT     #ILF,RMER1I     ;WAS "ILF" DETECTED??
12286  057636  001424                             BEQ     90$             ;NO!!
12287  057640  013737  001344  001140             MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12288  057646  042737  000001  001140             BIC     #ILF,$GDDAT
12289  057654  013737  001344  001142             MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12290  057662  062716  000004                     ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12291  057666  112776  000303  000000             MOVB    #303,@(SP)      ;WRITE ERROR NUMBER IN CALL
12292  057674  162716  000002                     SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12293  057700  004736                             JSR     PC,@(SP)+       ;REPORT ERORR AND RETURN
12294  057702  162716  000010                     SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12295  057706  000240                             NOP
12296  057710                           90$:
12297
12298                                   ;REPORT AN ERROR IF "RMR" IS SET
12299  057710  032737  000004  001344             BIT     #RMR,RMER1I     ;WAS "RMR" DETECTED??
12300  057716  001424                             BEQ     100$            ;NO!!
12301  057720  013737  001344  001140             MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12302  057726  042737  000004  001140             BIC     #RMR,$GDDAT
12303  057734  013737  001344  001142             MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12304  057742  062716  000004                     ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12305  057746  112776  000304  000000             MOVB    #304,@(SP)      ;WRITE ERROR NUMBER IN CALL
12306  057754  162716  000002                     SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12307  057760  004736                             JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12308  057762  162716  000010                     SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
```

```
12309  057766  000240                          NOP
12310  057770                          100$:
12311                                  ;DETERMINE WHETHER OR NOT "IAE" SHOULD BE SET AND CHECK FOR ERROR
12312  057770  012737  002000  001140          MOV     #IAE,$GDDAT     ;SETUP FOR "IAE" = 1
12313  057776  052737  040000  062432          BIS     #SKI,5OO$       ;SET SKI FLAG
12314  060004  022737  001466  001434          CMP     #822.,RMDCO     ;IS CYLINDER > 822??
12315  060012  103425                          BLO     11O$            ;YES!!
12316  060014  042737  040000  062432          BIC     #SKI,5OO$       ;RESET SKI FLAG
12317  060022  122737  000004  001407          CMPB    #4,RMDAO+1      ;IS TRACK > 4??
12318  060030  103416                          BLO     11O$            ;YES!!
12319  060032  122737  000037  001406          CMPB    #31.,RMDAO      ;IS SECTOR > 31??
12320  060040  103412                          BLO     11O$            ;YES!!
12321  060042  122737  000035  001406          CMPB    #29.,RMDAO      ;IS SECTOR > 29??
12322  060050  103004                          BHIS    105$            ;NO - IAE SHOULD BE ZERO
12323  060052  032737  010000  001432          BIT     #FMT16,RMOFO    ;18 BIT FORMAT??
12324  060060  001402                          BEQ     11O$            ;YES!!
12325  060062  005037  001140          105$:   CLR     $GDDAT          ;IAE SHOULD BE ZERO
12326  060066  013737  001344  001142  11O$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
12327  060074  042737  175777  001142          BIC     #↑CIAE,$BDDAT
12328  060102  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "IAE" STATUS OK??
12329  060110  001004                          BNE     115$            ;NO!!
12330  060112  042737  040000  062432          BIC     #SKI,5OO$       ;IAE OK - SKI SHOULD BE O
12331  060120  000412                          BR      12O$
12332  060122  062716  000004          115$:   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12333  060126  112776  000305  000000          MOVB    #3O5,@(SP)      ;WRITE ERROR NUMBER
12334  060134  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12335  060140  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12336  060142  162716  000010                  SUB     #1O,(SP)        ;MOVE SP TO NO ERROR
12337  060146                          12O$:
12338
12339                                  ;REPORT AN ERROR IF "SKI" IS SET AND "IAE" STATUS WAS OK
12340  060146  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12341  060154  042737  137777  001142          BIC     #↑CSKI,$BDDAT
12342  060162  013737  062432  001140          MOV     5OO$,$GDDAT     ;EXPECTED STATUS
12343  060170  042737  137777  001140          BIC     #↑CSKI,$GDDAT
12344  060176  032737  040000  001372          BIT     #SKI,RMER2I     ;WAS "SKI" SET??
12345  060204  001417                          BEQ     14O$            ;NO!!
12346  060206  032737  040000  062432          BIT     #SKI,5OO$       ;WAS SKI CAUSED BY IAE = O??
12347  060214  001032                          BNE     15O$            ;YES - DON'T REPORT SKI
12348  060216  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12349  060222  112776  000306  000000          MOVB    #3O6,@(SP)      ;WRITE ERROR NUMBER
12350  060230  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12351  060234  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12352  060236  162716  000010                  SUB     #1O,(SP)        ;MOVE SP TO NO ERROR
12353  060242  000417                          BR      15O$
12354
12355  060244                          14O$:
12356
12357                                  ;REPORT AN ERROR IF SKI = O AND IAE WAS NOT DETECTED
12358  060244  032737  040000  062432          BIT     #SKI,5OO$       ;SHOULD SKI BE SET??
12359  060252  001413                          BEQ     15O$            ;NO!!
12360  060254  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12361  060260  112776  000307  000000          MOVB    #3O7,@(SP)      ;WRITE ERROR NUMBER IN CALL
12362  060266  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12363  060272  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12364  060274  162716  000010                  SUB     #1O,(SP)        ;RESTORE SP TO NO ERROR
```

```
12365  060300  000240                          NOP
12366  060302                          150$:
12367
12368                                  ;LOOK FOR "LSC" OR "LBC" OR "DVC" IN ERROR REGISTER #2
12369  060302  032737  006200  001372          BIT     #LSC!LBC!DVC,RMER2I
12370  060310  001512                          BEQ     180$            ;NO ERRORS SET
12371
12372                                  ;REPORT ANY DEVICE FAULT, I.E., "DVC" = 1
12373  060312  032737  000200  001372          BIT     #DVC,RMER2I     ;IS "DVC" = 1??
12374  060320  001424                          BEQ     160$            ;NO!!
12375  060322  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12376  060330  042737  000200  001140          BIC     #DVC,$GDDAT
12377  060336  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12378  060344  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
12379  060350  112776  000310  000000          MOVB    #310,@(SP)      ;WRITE ERROR NUMBER IN CALL
12380  060356  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12381  060362  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12382  060364  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12383  060370  000240                          NOP
12384  060372                          160$:
12385
12386                                  ;REPORT LOSS OF BIT CLOCK, I.E.; "LBC" = 1, IF "MOL" = 1
12387  060372  032737  002000  001372          BIT     #LBC,RMER2I     ;IS LBC SET??
12388  060400  001430                          BEQ     170$            ;NO!!
12389  060402  032737  010000  001342          BIT     #MOL,RMDSI      ;WAS LBC ERROR BY MOL = 0
12390  060410  001424                          BEQ     170$            ;YES!!
12391  060412  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12392  060420  042737  002000  001140          BIC     #LBC,$GDDAT
12393  060426  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12394  060434  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12395  060440  112776  000311  000000          MOVB    #311,@(SP)      ;WRITE ERROR NUMBER IN CALL
12396  060446  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12397  060452  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12398  060454  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12399  060460  000240                          NOP
12400  060462                          170$:
12401
12402                                  ;REPORT LOS OF SYSTEM CLOCK, I.E., "LSC" = 1
12403  060462  032737  004000  001372          BIT     #LSC,RMER2I     ;IS "LSC" = 1??
12404  060470  001422                          BEQ     180$            ;NO!!
12405  060472  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12406  060500  042737  004000  001140          BIC     #LSC,$GDDAT
12407  060506  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12408  060514  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12409  060520  112776  000312  000000          MOVB    #312,@(SP)      ;WRITE ERROR NUMBER
12410  060526  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12411  060530  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12412  060534  000240                          NOP
12413  060536                          180$:
12414
12415                                  ;LOOK FOR "UNS" OR "DTE" OR "WLE" IN ERROR REGISTER #1
12416  060536  032737  054000  001344          BIT     #UNS!DTE!WLE,RMER1I
12417  060544  001527                          BEQ     220$            ;NO BITS SET
12418
12419  060546  032737  040000  001344  ;REPORT "UNS" ERROR IF "DVC" = 0
                                               BIT     #UNS,RMER1I     ;IS "UNS" SET??
12420  060554  001427                          BEQ     190$            ;NO!!
```

```
12421  060556  032737  000200  001372          BIT     #DVC,RMER2I     ;WAS "UNS" CAUSED BY "DVC"??
12422  060564  001023                          BNE     190$            ;YES!!
12423  060566  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12424  060574  042737  040000  001140          BIC     #UNS,$GDDAT
12425  060602  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12426  060610  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12427  060614  112776  000313  000000          MOVB    #313,@(SP)      ;WRITE ERROR NUMBER
12428  060622  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12429  060626  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12430  060630  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
12431  060634                          190$:
12432
12433
12434  060634  032737  010000  001344  ;REPORT ANY DRIVE TIMING ERROR, I.E. "DTE" = 1
12435  060642  001423                  BIT     #DTE,RMER1I     ;IS DTE SET??
12436  060644  013737  001344  001140  BEQ     200$            ;NO!!
12437  060652  042737  010000  001140  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12438  060660  013737  001344  001142  BIC     #DTE,$GDDAT
12439  060666  062716  000004          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12440  060672  112776  000314  000000  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12441  060700  162716  000002          MOVB    #314,@(SP)      ;WRITE ERROR NUMBER IN CALL
12442  060704  004736                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12443  060706  162716  000010          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12444  060712                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12445                          200$:
12446
12447  ;REPORT AN ERROR IF WRITE LOCK ERROR IS SET.  SEE IF DRIVE IS NOT
       ;WRITE PROTECTED, OR IF FUNCTION WAS NOT A WRITE
12448  060712  032737  004000  001344          BIT     #WLE,RMER1I     ;WAS "WLE" SET??
12449  060720  001441                          BEQ     220$            ;NO!!
12450  060722  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12451  060730  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12452  060736  052737  004000  001140          BIS     #WLE,$GDDAT
12453  060744  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12454  060750  112776  000315  000000          MOVB    #315,@(SP)      ;WRITE ERROR NUMBER IN CALL
12455  060756  032737  004000  001342          BIT     #WRL,RMDSI      ;WAS DRIVE WRITE PROTECTED??
12456  060764  001404                          BEQ     205$            ;NO!!
12457  060766  032737  000010  001400          BIT     #BIT3,RMCS10    ;WAS COMMAND A WRITE??
12458  060774  001406                          BEQ     210$            ;YES!!
12459  060776  112776  000316  000000  205$:   MOVB    #316,@(SP)      ;CHANGE ERROR NUMBER
12460  061004  042737  004000  001140          BIC     #WLE,$GDDAT
12461  061012  162716  000002          210$:   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12462  061016  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12463  061020  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12464
12465  061024                          220$:
12466
12467  ;OMIT DATA ERROR CHECKS IF ANY PREVIOUS ERRORS HAVE BEEN DETECTED
12468  061024  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12469  061030  105776  000000                  TSTB    @(SP)           ;WAS ERROR DETECTED??
12470  061034  001404                          BEQ     225$            ;NO - DO DATA CHECKS
12471  061036  162716  000004                  SUB     #4,(SP)         ;RESTORE SP
12472  061042  000137  062044                  JMP     340$            ;SKIP DATA CHECKS
12473  061046  162716  000004          225$:   SUB     #4,(SP)         ;RESTORE SP
12474
12475
12476  ;CHECK HEADER ERRORS IF FUNCTION WAS NOT WRITE HEADER AND DATA, AND
       ;IF HEADER COMPARE IS NOT INHIBITED
```

```
12477  061052  013737  001400  062434          MOV     RMCS10,S10$     ;STRIP AND STORE FUNCTION CODE
12478  061060  042737  177700  062434          BIC     #^CFNCMSK,S10$
12479  061066  022737  000063  062434          CMP     #WH!GO,S10$     ;WAS FUNCTION WRITE HEADER & DATA??
12480  061074  001512                          BEQ     250$            ;YES - SKIP HEADER CHECKS
12481  061076  032737  002000  001362          BIT     #HCI,RMOFI      ;WAS HCI SET??
12482  061104  001106                          BNE     250$            ;YES - SKIP HEADER CHECKS
12483
12484                                  ;SEE IF ANY HEADER ERRORS ARE SET, I.E., "FER" OR "HCRC" OR "HCE"
12485  061106  032737  000620  001344          BIT     #HCRC!FER!HCE,RMER1I
12486  061114  001533                          BEQ     270$            ;NO ERRORS SET
12487
12488                                  ;REPORT HEADER CRC ERROR IF SET
12489  061116  032737  000400  001344          BIT     #HCRC,RMER1I    ;WAS HCRC SET??
12490  061124  001422                          BEQ     230$            ;NO!!
12491  061126  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12492  061134  042737  000400  001140          BIC     #HCRC,$GDDAT
12493  061142  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12494  061150  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
12495  061154  112776  000317  000000          MOVB    #317,@(SP)      ;WRITE ERROR NUMBER
12496  061162  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12497  061166  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12498  061170  000501                          BR      260$
12499  061172                          230$:
12500
12501                                  ;REPORT FORMAT ERROR IF SET
12502  061172  032737  000020  001344          BIT     #FER,RMER1I     ;WAS "FER" SET??
12503  061200  001422                          BEQ     240$            ;NO!!
12504  061202  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12505  061210  042737  000020  001140          BIC     #FER,$GDDAT
12506  061216  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12507  061224  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
12508  061230  112776  000320  000000          MOVB    #320,@(SP)      ;WRITE ERROR NUMBER
12509  061236  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12510  061242  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12511  061244  000453                          BR      260$
12512  061246                          240$:
12513
12514                                  ;REPORT HEADER COMPARE ERROR IF SET
12515  061246  032737  000200  001344          BIT     #HCE,RMER1I     ;WAS "HCE" SET??
12516  061254  001453                          BEQ     270$            ;NO!!
12517  061256  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12518  061264  042737  000200  001140          BIC     #HCE,$GDDAT
12519  061272  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12520  061300  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12521  061304  112776  000321  000000          MOVB    #321,@(SP)      ;WRITE ERROR NUMBER
12522  061312  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12523  061316  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12524  061320  000425                          BR      260$
12525                                  ;THERE SHOULD BE NO HEADER ERRORS BECAUSE
12526                                  ;       .COMMAND WAS WRITE HEADER AND DATA, OR
12527                                  ;       .HEADER COMPARE INHIBIT WAS SET
12528  061322  032737  000620  001344  250$:   BIT     #HCE!FER!HCRC,RMER1I
12529  061330  001425                          BEQ     270$            ;NO ERRORS WERE SET
12530  061332  013737  001344  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12531  061340  042737  000620  001140          BIC     #HCE!FER!HCRC,$GDDAT
12532  061346  013737  001344  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
```

```
12533  061354  062716  000004                ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
12534  061360  112776  000322  000000        MOVB    #322,@(SP)          ;WRITE ERROR NUMBER
12535  061366  162716  000002                SUB     #2,(SP)             ;MOVE SP TO RETURN IF ERROR
12536  061372  004736                        JSR     PC,@(SP)+           ;REPORT ERROR AND RETURN
12537  061374  162716  000010        260$:   SUB     #10,(SP)            ;MOVE SP TO NO ERROR
12538  061400  000137  062044                JMP     340$                ;OMIT FURTHER DATA CHECKS
12539
12540  061404                        270$:
12541
12542                                ;IF COMMAND WAS A WRITE COMMAND, GO DO WRITE ERROR CHECKS, OTHERWISE
12543                                ;DO READ ERROR CHECKS
12544  061404  032737  000010  062434        BIT     #BIT3,510$          ;WAS THIS A WRITE COMMAND?
12545  061412  001002                        BNE     275$                ;NO!!
12546  061414  000137  061632                JMP     310$                ;GO DO WRITE STATUS CHECK
12547  061420                        275$:
12548
12549                                ;REPORT DATA CHECK IF SET
12550  061420  032737  100000  001344        BIT     #DCK,RMER1I         ;DATA CHECK ERROR??
12551  061426  001450                        BEQ     290$                ;NO!!
12552  061430  013737  001344  001140        MOV     RMER1I,$GDDAT       ;EXPECTED STATUS
12553  061436  042737  100000  001140        BIC     #DCK,$GDDAT
12554  061444  013737  001344  001142        MOV     RMER1I,$BDDAT       ;RECEIVED STATUS
12555  061452  062716  000004                ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR
12556  061456  112776  000323  000000        MOVB    #323,@(SP)          ;WRITE ERROR NUMBER
12557  061464  032737  004000  001362        BIT     #ECI,RMOFI          ;WAS ECC CORRECTION DISABLED??
12558  061472  001021                        BNE     280$                ;YES!!
12559  061474  112776  000324  000000        MOVB    #324,@(SP)          ;CHANGE TO RECOVERABLE ERROR
12560  061502  032737  000100  001344        BIT     #ECH,RMER1I         ;IS ERROR RECOVERABLE??
12561  061510  001007                        BNE     276$                ;NO !!
12562
12563  061512  032737  000020  062434        BIT     #BIT4,510$          ;WAS THIS A READ COMMAND ??
12564  061520  001406                        BEQ     280$                ;NO !!
12565  061522  162716  000004                SUB     #4,(SP)             ;RESTORE SP
12566  061526  000410                        BR      290$                ;SKIP ERROR - DATA WILL BE CORRECTED
12567  061530  112776  000325  000000  276$: MOVB    #325,@(SP)          ;CHANGE TO NON RECOVERABLE
12568  061536  162716  000002        280$:   SUB     #2,(SP)             ;MOVE SP TO RETURN IF ERROR
12569  061542  004736                        JSR     PC,@(SP)+           ;REPORT ERROR AND RETURN
12570  061544  162716  000010                SUB     #10,(SP)            ;RESTORE SP TO NO ERROR
12571
12572  061550                        290$:
12573
12574                                ;REPORT DATA BUS PARITY ERROR IF SET, I.E., MDPE = 1
12575  061550  032737  000400  001340        BIT     #MDPE,RMCS2I        ;PARITY ERROR SET??
12576  061556  001423                        BEQ     300$                ;NO!!
12577  061560  013737  001340  001140        MOV     RMCS2I,$GDDAT       ;EXPECTED STATUS
12578  061566  042737  000400  001140        BIC     #MDPE,$GDDAT
12579  061574  013737  001340  001142        MOV     RMCS2I,$BDDAT       ;RECEIVED STATUS
12580  061602  062716  000004                ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR
12581  061606  112776  000326  000000        MOVB    #326,@(SP)          ;WRITE ERROR NUMBER
12582  061614  162716  000002                SUB     #2,(SP)             ;MOVE SP TO RETURN IF ERROR
12583  061620  004736                        JSR     PC,@(SP)+           ;REPORT ERROR AND RETURN
12584  061622  162716  000010                SUB     #10,(SP)            ;MOVE SP TO NO ERROR
12585  061626  000137  062044        300$:   JMP     340$                ;SKIP WRITE STATUS CHECK
12586
12587  061632                        310$:
12588
```

```
12589
12590  061632  032737  000001  001342          ;TEST TO SEE THAT OFFSET MODE WAS RESET; REPORT ERROR IF "OM" = 1
12591  061640  001423                  BIT     #OM,RMDSI       ;IS OFFSET ON??
12592  061642  013737  001342  001140  BEQ     320$            ;NO
12593  061650  042737  000001  001140  MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12594  061656  013737  001342  001142  BIC     #OM,$GDDAT
12595  061664  062716  000004          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12596  061670  112776  000327  000000  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12597  061676  162716  000002          MOVB    #327,@(SP)      ;WRITE ERROR NUMBER IN CALL
12598  061702  004736                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12599  061704  162716  000010          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12600  061710                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12601                          320$:
12602                          ;TEST FOR DATA BUS PARITY ERROR; REPORT ERROR IF "DPE" = 1
12603  061710  032737  000010  001372          BIT     #DPE,RMER2I     ;DATA PARITY ERROR??
12604  061716  001423                  BEQ     330$            ;NO!!
12605  061720  013737  001372  001140  MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12606  061726  042737  000010  001140  BIC     #DPE,$GDDAT
12607  061734  013737  001372  001142  MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12608  061742  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12609  061746  112776  000330  000000  MOVB    #330,@(SP)      ;WRITE ERROR NUMBER
12610  061754  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12611  061760  004736                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12612  061762  162716  000010          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12613  061766                          330$:
12614
12615                          ;TEST FOR WRITE CLOCK FAILURE; REPORT ERROR IF "WCF" = 1
12616  061766  032737  000040  001344          BIT     #WCF,RMER1I     ;IS "WCF" SET??
12617  061774  001423                  BEQ     340$            ;NO!!
12618  061776  013737  001344  001140  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
12619  062004  042737  000040  001140  BIC     #WCF,$GDDAT
12620  062012  013737  001344  001142  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
12621  062020  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12622  062024  112776  000331  000000  MOVB    #331,@(SP)      ;WRITE ERROR NUMBER
12623  062032  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12624  062036  004736                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12625  062040  162716  000010          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12626  062044                          340$:
12627
12628                          ;REPORT "DATA LATE" ERROR IF "DLT" = 1
12629  062044  032737  100000  001340          BIT     #DLT,RMCS2I     ;IS "DLT" SET??
12630  062052  001423                  BEQ     350$            ;NO!!
12631  062054  013737  001340  001140  MOV     RMCS2I,$GDDAT   ;EXPECTED STATUS
12632  062062  042737  100000  001140  BIC     #DLT,$GDDAT
12633  062070  013737  001340  001142  MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
12634  062076  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12635  062102  112776  000332  000000  MOVB    #332,@(SP)      ;WRITE ERROR NUMBER
12636  062110  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12637  062114  004736                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12638  062116  162716  000010          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12639  062122                          350$:
12640                          ;LOOK FOR UNEXPECTED CHANGES IN DRIVE STATUS
12641  062122  013746  001342          MOV     RMDSI,-(SP)     ;STACK DRIVE STATUS
12642  062126  042716  147677          BIC     #^C<PIP!MOL!VV>,(SP) ;CLEAR DONT CARES
12643  062132  022726  010100          CMP     #MOL!VV,(SP)+   ;IS DRIVE STATUS OK??
12644  062136  001522                  BEQ     380$            ;YES!!
```

```
12645
12646                                   ;REPORT ERROR IF POSITIONING IN PROGRESS AND NO SEEK INCOMPLETE ERROR,
12647                                   ;I.E, PIP = 1 AND SKI = 0
12648  062140  032737  020000  001342          BIT     #PIP,RMDSI      ;IS "PIP" SET??
12649  062146  001430                        BEQ     360$            ;NO!!
12650  062150  032737  040000  001372        BIT     #SKI,RMER2I     ;WAS "SKI" ERROR REPORTED??
12651  062156  001024                        BNE     360$            ;YES-DONT REPORT PIP
12652  062160  013737  001342  001140        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12653  062166  042737  020000  001140        BIC     #PIP,$GDDAT
12654  062174  013737  001342  001142        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12655  062202  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12656  062206  112776  000333  000000        MOVB    #333,@(SP)      ;WRITE ERROR NUMBER
12657  062214  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12658  062220  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12659  062222  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12660  062226  000240                        NOP
12661  062230                          360$:
12662
12663
12664                                   ;REPORT ERROR IF MEDIUM IS NOT ON LINE AND OPI ERROR WAS NOT
12665  062230  032737  010000  001342  ;REPORTED, I.E., MOL = OPI = 0
12666  062236  001027                        BIT     #MOL,RMDSI      ;IS MEDIUM ON LINE??
12667  062240  032737  020000  001344        BNE     370$            ;YES!!
12668  062246  001023                        BIT     #OPI,RMER1I     ;WAS OPI ERROR REPORTED??
12669  062250  013737  001342  001140        BNE     370$            ;YES!!
12670  062256  052737  010000  001140        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12671  062264  013737  001342  001142        BIS     #MOL,$GDDAT
12672  062272  062716  000004                MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12673  062276  112776  000334  000000        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
12674  062304  162716  000002                MOVB    #334,@(SP)      ;WRITE ERROR NUMBER
12675  062310  004736                        SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12676  062312  162716  000010                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12677  062316                          370$:   SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12678
12679                                   ;REPORT ERROR IF VOLUME IS NOT VALID AND "IVC" ERROR WAS NOT
12680                                   ;REPORTED, I.E., VV = IVC = 0
12681  062316  032737  000100  001342          BIT     #VV,RMDSI       ;IS VOLUME VALID??
12682  062324  001027                        BNE     380$            ;YES!!
12683  062326  032737  010000  001372        BIT     #IVC,RMER2I     ;WAS IVC ERROR REPORTED??
12684  062334  001033                        BNE     390$            ;YES!!
12685  062336  013737  001342  001140        MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12686  062344  052737  000100  001140        BIS     #VV,$GDDAT
12687  062352  013737  001342  001142        MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12688  062360  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
12689  062364  112776  000335  000000        MOVB    #335,@(SP)      ;WRITE ERROR NUMBER
12690  062372  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
12691  062376  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
12692  062400  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
12693  062404                          380$:
12694
12695                                   ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS FOUND
12696  062404  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
12697  062410  105776  000000                TSTB    @(SP)           ;ANY ERROR??
12698  062414  001403                        BEQ     390$            ;NO!!
12699  062416  062716  000004                ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
12700  062422  000402                        BR      400$
```

```
12701  062424  162716  000004          390$:   SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
12702
12703  062430  000207                  400$:   RTS     PC              ;RETURN TO USER
12704
12705  062432  000000                  500$:   .WORD                   ;ERROR FLAGS
12706  062434  000000                  510$:   .WORD                   ;TEMPORARY STORAGE
```

```
12707                                   .SBTTL  STATIC DRIVE STATUS CHECK SUBROUTINE
12708
12709                                   ;THIS SUBROUTINE LOOKS FOR UNEXPECTED CHANGES IN DRIVE
12710                                   ;STATUS, SUCH AS THE DRIVE LOSING VOLUME VALID.  THE SUBROUTINE
12711                                   ;CAN BE USED BY HOUSEKEEPING AND OTHER COMMANDS DURING WHICH THERE
12712                                   ;SHOULD NOT BE ANY DRIVE ERRORS OR CHANGES IN STATE.
12713
12714                                   ;THE FOLLOWING CONDITIONS ARE TESTED AND REPORTED AS ERRORS
12715                                   ;IF TRUE:
12716
12717                                   ;       .MOL = 0, INDICATES DRIVE WENT OFFLINE, NOTE
12718                                   ;THAT MOL IS ASSUMED TO HAVE BEEN SET
12719                                   ;       .VV = 0, INDICATES THE DRIVE LOST VOLUME VALID
12720                                   ;       .PIP = 1, INDICATES THAT THE DRIVE IS OFF CYLINDER
12721                                   ;       .SKI = 1, INDICATES THE DRIVE HAS AN UNEXPECTED SKI ERROR
12722                                   ;       .DVC = 1, INDICATES AN UNEXPECTED DEVICE FAULT
12723
12724                                   ;THE SUBROUTINE IS CALLED AFTER STORING STATUS IN THE GET BUFFER.
12725
12726                                   ;(1)    JSR     PC,STCDRVSTS
12727                                   ;       BR      ???             RETURN HERE IF NO ERROR
12728                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
12729                                   ;       .ERROR                  ERROR NUMBER DEFINED BY SUB
12730                                   ;       JMP     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
12731                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
12732
12733  062436                           STCDRVSTS:
12734
12735                                   ;CLEAR USER'S ERROR CALL
12736  062436  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
12737  062442  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
12738  062446  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR RETURN
12739
12740  062452  013746  001342          ;SEE IF "MOL" = "VV" = 1, AND "PIP" = 0
12741  062456  042716  147677                  MOV     RMDSI,-(SP)     ;PUT DRIVE STATUS ON STACK
12742  062462  022726  010100                  BIC     #↑C<PIP!MOL!VV>,(SP)
12743  062466  001524                          CMP     #MOL!VV,(SP)+   ;ARE MOL,VV AND PIP O.K.??
12744                                           BEQ     30$             ;YES!!
12745
12746  062470  032737  010000  001342  ;REPORT AN ERROR IF MOL = 0 AND "OPI" = 0
12747  062476  001030                          BIT     #MOL,RMDSI      ;IS MOL ON ??
12748  062500  032737  020000  001344          BNE     10$             ;YES!!
12749  062506  001024                          BIT     #OPI,RMER1I     ;WAS "OPI" SET??
12750  062510  013737  001342  001140          BNE     10$             ;YES-DONT REPORT "MOL" = 0
12751  062516  052737  010000  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12752  062524  013737  001342  001142          BIS     #MOL,$GDDAT
12753  062532  062716  000004                  MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12754  062536  112776  000207  000000          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12755  062544  162716  000002                  MOVB    #207,@(SP)      ;WRITE ERROR NUMBER IN CALL
12756  062550  004736                          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12757  062552  162716  000010                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
                                               SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
```

```
12758  062556  000240                          NOP
12759  062560                          10$:
12760
12761                                  ;REPORT AN ERROR IF VOLUME VALID IS NOW ZERO AND "IVC" = 0
12762  062560  032737  000100  001342          BIT     #VV,RMDSI       ;IS "VV" = 0??
12763  062566  001030                          BNE     20$             ;NO!!
12764  062570  032737  010000  001372          BIT     #IVC,RMER2I     ;WAS "IVC" SET??
12765  062576  001024                          BNE     20$             ;YES-DONT REPORT "VV" = 0
12766  062600  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12767  062606  052737  000100  001342          BIS     #VV,RMDSI
12768  062614  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12769  062622  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
12770  062626  112776  000210  000000          MOVB    #210,@(SP)      ;WRITE ERROR NUMBER IN CALL
12771  062634  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12772  062640  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12773  062642  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
12774  062646  000240                          NOP
12775  062650                          20$:
12776
12777                                  ;REPORT AN ERROR IF DRIVE IS OFF CYLINDER AND "SKI" = 0
12778  062650  032737  020000  001342          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
12779  062656  001430                          BEQ     30$             ;NO!!
12780  062660  032737  040000  001372          BIT     #SKI,RMER2I     ;WAS "SKI" SET??
12781  062666  001024                          BNE     30$             ;YES-DONT REPORT "PIP" = 1
12782  062670  013737  001342  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
12783  062676  042737  020000  001140          BIC     #PIP,$GDDAT
12784  062704  013737  001342  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
12785  062712  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12786  062716  112776  000211  000000          MOVB    #211,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
12787  062724  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12788  062730  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12789  062732  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12790  062736  000240                          NOP
12791  062740                          30$:
12792
12793                                  ;SEE IF "SKI" = "DVC" = 0
12794  062740  013746  001372                  MOV     RMER2I,-(SP)    ;PUT ERROR REG 2 ON STACK
12795  062744  042726  137577                  BIC     #^C<SKI!DVC>,(SP)+
12796  062750  001460                          BEQ     60$             ;BRANCH IF NO ERROR
12797  062752                          40$:
12798
12799                                  ;REPORT AN ERROR IF THERE IS A DEVICE FAULT
12800  062752  032737  000200  001372          BIT     #DVC,RMER2I     ;ANY DEVICE FAULT??
12801  062760  001424                          BEQ     50$             ;NO!!
12802  062762  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12803  062770  042737  000200  001140          BIC     #DVC,$GDDAT
12804  062776  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12805  063004  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S CALL
12806  063010  112776  000212  000000          MOVB    #212,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
12807  063016  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12808  063022  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12809  063024  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
12810  063030  000240                          NOP
12811  063032                          50$:
12812
12813                                  ;REPORT AN ERROR IF "SKI" = 1
```

```
12814  063032  032737  040000  001372          BIT     #SKI,RMER2I     ;IS THERE A SEEK INCOMPLETE ERROR
12815  063040  001424                          BEQ     60$             ;NO!!
12816  063042  013737  001372  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
12817  063050  042737  040000  001140          BIC     #SKI,$GDDAT
12818  063056  013737  001372  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
12819  063064  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12820  063070  112776  000213  000000          MOVB    #213,@(SP)      ;WRITE ERROR NUMBER IN USER'S ERROR CALL
12821  063076  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12822  063102  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12823  063104  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
12824  063110  000240                          NOP
12825  063112                          60$:
12826
12827                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
12828  063112  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12829  063116  105776  000000                  TSTB    @(SP)           ;WAS AN ERROR DETECTED??
12830  063122  001403                          BEQ     70$             ;NO!!
12831  063124  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO USER'S ERROR RETURN
12832  063130  000402                          BR      80$
12833  063132  162716  000004          70$:    SUB     #4,(SP)         ;NO - MOVE SP TO NO ERROR RETURN
12834  063136  000240                  80$:    NOP
12835  063140  000207                          RTS     PC              ;RETURN TO USER
```

```
12836                                   .SBTTL  COMPOSITE ERROR CHECK SUBROUTINE
12837
12838                                   ;THIS SUBROUTINE CHECKS THE STORED CONTENTS OF RMER1 AND
12839                                   ;RMER2 AFTER MASKING EACH REGISTER WORD WITH THE USER'S STATUS
12840                                   ;MASKS AND REPORTS AN ERROR IF ANY BITS ARE LEFT ON AFTER
12841                                   ;THE MASKS ARE APPLIED.
12842
12843                                   ;CALL:
12844                                   ;(1)    JSR     PC,CMPERRSTS
12845                                   ;       .WORD                   MASK FOR ERROR REGISTER 1
12846                                   ;       .WORD                   MASK FOR ERROR REGISTER 2
12847                                   ;       BR      ???             RETURN HERE IF NO ERROR
12848                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
12849                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
12850                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
12851                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
12852
12853                                   ;NOTE:  BITS TO BE MASKED SHOULD BE ONE; BITS TO BE TESTED SHOULD
12854                                   ;BE ZERO
12855
12856   063142                          CMPERRSTS:
12857
12858                                   ;MASK AND STORE THE CONTENTS OF RMER1  AND RMER2
12859   063142  013737  001344  001176          MOV     RMER1I,$TMP1    ;STORE RMER1 AT TEMP STORAGE
12860   063150  047637  000000  001176          BIC     @(SP),$TMP1     ;MASK   RMER1
12861   063156  062716  000002                  ADD     #2,(SP)         ;MOVE SP TO NEXT MASK
12862   063162  013737  001372  001200          MOV     RMER2I,$TMP2    ;STORE RMER2 AT TEMP STORAGE
12863   063170  047637  000000  001200          BIC     @(SP),$TMP2     ;MASK RMER2
12864
12865
12866                                   ;CLEAR USER'S ERROR CALL
12867   063176  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USER'S ERROR CALL
12868   063202  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
12869   063206  162716  000004                  SUB     #4,(SP)         ;LEAVE SP AT NO ERROR RETURN
12870
12871                                   ;SEE IF THERE WERE ANY ERRORS IN RMER1, I.E., $TMP1
12872   063212  005737  001176                  TST     $TMP1           ;ANY ERRORS TO REPORT??
12873   063216  001420                          BEQ     5$              ;NO !!
12874   063220  013737  001176  001142          MOV     $TMP1,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
12875   063226  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
12876   063232  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12877   063236  112776  000066  000000          MOVB    #66,@(SP)       ;CORRECTABLE DATA CHECK ERROR #
12878   063244  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12879   063250  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12880   063252  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
12881   063256  000240                          NOP
12882   063260                          5$:
12883
12884
12885                                   ;SEE IF THERE ARE ANY ERRORS TO REPORT IN RMER2 ($TMP2)
12886   063260  005737  001200                  TST     $TMP2           ;ANY ERRORS IN RMER2?
12887   063264  001420                          BEQ     10$             ;NO!!
12888
12889   063266  013737  001200  001142          MOV     $TMP2,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
12890   063274  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
12891   063300  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
```

```
12892  063304  112776  000067  000000          MOVB    #67,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
12893  063312  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
12894  063316  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
12895  063320  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
12896  063324  000240                          NOP
12897  063326                          10$:
12898
12899                          ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
12900  063326  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
12901  063332  105776  000000                  TSTB    @(SP)           ;WAS THERE AN ERROR CALLED??
12902  063336  001403                          BEQ     20$             ;NO!!
12903  063340  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
12904  063344  000402                          BR      30$
12905  063346  162716  000004          20$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
12906  063352  000207                  30$:    RTS     PC              ;RETURN TO USER
12907
12908
```

```
12909                                   .SBTTL  STOP AND SHUTDOWN SUBROUTINES
12910
12911   063354                          STOP:
12912
12913                                   ;DROP PRIORITY TO ALLOW CONSOLE INTERRUPT
12914   063354  012746  000140                  MOV     #PR3,-(SP)      ;;PUT NEW PS ON STACK
12915   063360  012746  063366                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
12916   063364  000002                          RTI                     ;;POP NEW PC AND PS
12917   063366                          64$:
12918   063366  000240                          NOP
12919                                   ;RAISE PRIORITY TO INHIBIT INTERRUPT
12920   063370  012746  000300                  MOV     #PR6,-(SP)      ;;PUT NEW PS ON STACK
12921   063374  012746  063402                  MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
12922   063400  000002                          RTI                     ;;POP NEW PC AND PS
12923   063402                          65$:
12924
12925   063402  000207                  10$:    RTS     PC              ;CONTINUE
12926
12927
12928   063404                          SHUT:
12929   063404  005737  001326                  TST     @#CTLFG         ;HALT ?
12930   063410  001002                          BNE     .+6             ;BRANCH IF SO
12931   063412  000137  007100                  JMP     READY           ;OTHERWISE LOOP
12932   063416  104401  063440                  TYPE    ,100$           ;TYPE THE HALT MESSAGE
12933   063422  005737  000042                  TST     42              ;RUNNING STANDALONE ??
12934   063426  001402                          BEQ     10$             ;YES !!
12935   063430  000137  037006                  JMP     $EOP            ;NO - GO TO END OF PASS
12936   063434                          10$:
12937   063434  000137  005324                  JMP     START           ;GO TO START
12938   063440  042524  052123  053440  100$:   .ASCIZ  @TEST WAS HALTED@<CR><LF><LF>
12939   063446  051501  044040  046101
12940   063454  042524  006504  005012
12941   063462     000
12942              063464                       .EVEN
```

```
12943  063464  012737  177777  001326  SHUT2:  MOV     #-1,CTLFG
12944  063472  000002                          RTI                     ;SET CONTROL C FLAG
12945                                  .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
12946
12947                                  ;;*********************************************************************
12948                                  ;*SAVE R0-R5
12949                                  ;*CALL:
12950                                  ;*      SAVREG
12951                                  ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
12952                                  ;*
12953                                  ;*TOP---(+16)
12954                                  ;* +2---(+18)
12955                                  ;* +4---R5
12956                                  ;* +6---R4
12957                                  ;* +8---R3
12958                                  ;*+10---R2
12959                                  ;*+12---R1
12960                                  ;*+14---R0
12961
12962  063474                          $SAVREG:
12963  063474  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12964  063476  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12965  063500  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12966  063502  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
12967  063504  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
12968  063506  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
12969  063510  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
12970  063514  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
12971  063520  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
12972  063524  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
12973  063530  000002                          RTI
12974
12975                                  ;*RESTORE R0-R5
12976                                  ;*CALL:
12977                                  ;*      RESREG
12978  063532                          $RESREG:
12979  063532  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
12980  063536  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
12981  063542  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
12982  063546  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
12983  063552  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
12984  063554  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
12985  063556  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
12986  063560  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
12987  063562  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
12988  063564  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
12989  063566  000002                          RTI
12990                                  .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
12991
12992                                  ;;*********************************************************************
12993                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
12994                                  ;*BINARY-ASCII NUMBER AND TYPE IT.
12995                                  ;*CALL:
12996                                  ;*      MOV     NUMBER,-(SP)    ;;NUMBER TO BE TYPED
12997                                  ;*      TYPBN                   ;;TYPE IT
12998
```

```
12999  063570  010146                  $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
13000  063572  016601  000006                  MOV     6(SP),R1        ;;GET THE INPUT NUMBER
13001  063576  000261                          SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
13002  063600  112737  000060  063642  1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
13003  063606  006101                          ROL     R1              ;;GET THIS BIT
13004  063610  001406                          BEQ     2$              ;;DONE?
13005  063612  105537  063642                  ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
13006  063616  104401  063642                  TYPE    ,$BIN           ;;GO TYPE THIS BIT
13007  063622  000241                          CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
13008  063624  000765                          BR      1$              ;;GO DO THE NEXT BIT
13009  063626  012601                  2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
13010  063630  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
13011  063636  012616                          MOV     (SP)+,(SP)
13012  063640  000002                          RTI                     ;;RETURN TO USER
13013  063642      000      000         $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
13014                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
13015
13016                                  ;;**************************************************************
13017                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
13018                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
13019                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
13020                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
13021                                  ;*REPLACED WITH SPACES.
13022                                  ;*CALL:
13023                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
13024                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
13025
13026  063644                          $TYPDS:
13027  063644  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
13028  063646  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
13029  063650  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
13030  063652  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
13031  063654  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
13032  063656  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
13033  063662  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
13034  063666  100004                          BPL     1$              ;;BR IF INPUT IS POS.
13035  063670  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
13036  063672  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
13037  063700  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
13038  063702  012703  064060                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
13039  063706  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
13040  063712  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
13041  063714  016001  064050                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
13042  063720  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
13043  063722  002402                          BLT     4$              ;;BR IF DONE
13044  063724  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
13045  063726  000774                          BR      3$
13046  063730  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
13047  063732  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
13048  063734  001002                          BNE     5$              ;;FALL THROUGH IF 0
13049  063736  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
13050  063740  100407                          BMI     7$              ;;BR IF YES
13051  063742  106316                  5$:     ASLB    (SP)            ;;MSD?
13052  063744  103003                          BCC     6$              ;;BR IF NO
13053  063746  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
13054  063754  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
```

```
13055  063760  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
13056  063764  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
13057  063766  005720                          TST     (R0)+           ;;JUST INCREMENTING
13058  063770  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
13059  063774  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
13060  063776  003002                          BGT     8$              ;;GO TO EXIT
13061  064000  010502                          MOV     R5,R2           ;;GET THE LSD
13062  064002  000764                          BR      6$              ;;GO CHANGE TO ASCII
13063  064004  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
13064  064006  100003                          BPL     9$              ;;BR IF NO
13065  064010  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
13066  064016  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
13067  064020  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
13068  064022  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
13069  064024  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
13070  064026  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
13071  064030  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
13072  064032  104401  064060                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
13073  064036  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
13074  064044  012616                          MOV     (SP)+,(SP)
13075  064046  000002                          RTI                     ;;RETURN TO USER
13076  064050  023420                  $DTBL:  10000.
13077  064052  001750                          1000.
13078  064054  000144                          100.
13079  064056  000012                          10.
13080  064060  000004                  $DBLK:  .BLKW   4
13081                                   .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
13082
13083                                   ;;**********************************************************************
13084                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
13085                                   ;*OCTAL (ASCII) NUMBER AND TYPE IT.
13086                                   ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
13087                                   ;*CALL:
13088                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
13089                                   ;*      TYPOS                   ;;CALL FOR TYPEOUT
13090                                   ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
13091                                   ;*      .BYTE   M               ;;M=1 OR 0
13092                                   ;*                                      ;;1=TYPE LEADING ZEROS
13093                                   ;*                                      ;;0=SUPPRESS LEADING ZEROS
13094                                   ;*
13095                                   ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
13096                                   ;*$TYPOS OR $TYPOC
13097                                   ;*CALL:
13098                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
13099                                   ;*      TYPON                   ;;CALL FOR TYPEOUT
13100                                   ;*
13101                                   ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
13102                                   ;*CALL:
13103                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
13104                                   ;*      TYPOC                   ;;CALL FOR TYPEOUT
13105
13106  064070  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
13107  064074  116637  000001  064313          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
13108  064102  112637  064315                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
13109  064106  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
13110  064112  000406                          BR      $TYPON
```

```
13111  064114  112737  000001  064313  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
13112  064122  112737  000006  064315          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
13113  064130  112737  000005  064312  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
13114  064136  010346                          MOV     R3,-(SP)        ;;SAVE R3
13115  064140  010446                          MOV     R4,-(SP)        ;;SAVE R4
13116  064142  010546                          MOV     R5,-(SP)        ;;SAVE R5
13117  064144  113704  064315                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
13118  064150  005404                          NEG     R4
13119  064152  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
13120  064156  110437  064314                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
13121  064162  113704  064313                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
13122  064166  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
13123  064172  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
13124  064174  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
13125  064176  000404                          BR      3$              ;;GO DO MSB
13126  064200  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
13127  064202  006105                          ROL     R5
13128  064204  006105                          ROL     R5
13129  064206  010503                          MOV     R5,R3
13130  064210  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
13131  064212  105337  064314                  DECB    $OMODE          ;;TYPE THIS DIGIT?
13132  064216  100016                          BPL     7$              ;;BR IF NO
13133  064220  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
13134  064224  001002                          BNE     4$              ;;TEST FOR 0
13135  064226  005704                          TST     R4              ;;SUPPRESS THIS 0?
13136  064230  001403                          BEQ     5$              ;;BR IF YES
13137  064232  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
13138  064234  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
13139  064240  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
13140  064244  110337  064310                  MOVB    R3,8$           ;;SAVE FOR TYPING
13141  064250  104401  064310                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
13142  064254  105337  064312          7$:     DECB    $OCNT           ;;COUNT BY 1
13143  064260  003347                          BGT     2$              ;;BR IF MORE TO DO
13144  064262  002402                          BLT     6$              ;;BR IF DONE
13145  064264  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
13146  064266  000744                          BR      2$              ;;GO DO THE LAST DIGIT
13147  064270  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
13148  064272  012604                          MOV     (SP)+,R4        ;;RESTORE R4
13149  064274  012603                          MOV     (SP)+,R3        ;;RESTORE R3
13150  064276  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
13151  064304  012616                          MOV     (SP)+,(SP)
13152  064306  000002                          RTI                     ;;RETURN
13153  064310     000          8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
13154  064311     000                  .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
13155  064312     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
13156  064313     000          $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
13157  064314  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
13158                          .SBTTL  TYPE ROUTINE
13159
13160                          ;;******************************************************************
13161                          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
13162                          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
13163                          ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
13164                          ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
13165                          ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
13166                          ;*
```

```
13167                                   ;*CALL:
13168                                   ;*1) USING A TRAP INSTRUCTION
13169                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
13170                                   ;*OR
13171                                   ;*      TYPE
13172                                   ;*      MESADR
13173                                   ;*
13174
13175  064316  105737  001173          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
13176  064322  100002                          BPL     1$              ;;BR IF YES
13177  064324  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
13178  064326  000430                          BR      3$              ;;LEAVE
13179  064330  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
13180  064332  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
13181  064336  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
13182  064344  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
13183  064346  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
13184  064354  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
13185  064356  010037  064366                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
13186  064362  004737  067226                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
13187  064366  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
13188  064370  132737  000040  001243  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
13189  064376  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
13190  064400  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
13191  064402  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
13192  064404  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
13193  064406  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
13194  064410  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
13195  064414  000002                          RTI                     ;;RETURN
13196  064416  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
13197  064422  001430                          BEQ     8$
13198  064424  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
13199  064430  001006                          BNE     5$
13200  064432  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
13201  064434  104401                          TYPE                    ;;TYPE A CR AND LF
13202  064436  001217                          $CRLF
13203  064440  105037  064574                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
13204  064444  000755                          BR      2$              ;;GET NEXT CHARACTER
13205  064446  004737  064530          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
13206  064452  123726  001172          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
13207  064456  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
13208  064460  013746  001170                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
13209                                          ;;AND THE NULL CHAR.
13210  064464  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
13211  064470  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
13212  064472  004737  064530                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
13213  064476  105337  064574                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
13214  064502  000770                          BR      7$              ;;LOOP
13215
13216                                   ;HORIZONTAL TAB PROCESSOR
13217
13218  064504  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
13219  064510  004737  064530          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
13220  064514  132737  000007  064574          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
13221  064522  001372                          BNE     9$              ;;TAB STOP
13222  064524  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
```

```
13223  064526  000724                          BR      2$                      ;;GET NEXT CHARACTER
13224  064530  105777  114430          $TYPEC: TSTB    @$TPS                   ;;WAIT UNTIL PRINTER IS READY
13225  064534  100375                          BPL     $TYPEC
13226  064536  116677  000002  114422          MOVB    2(SP),@$TPB             ;;LOAD CHAR TO BE TYPED INTO DATA REG.
13227  064544  122766  000015  000002          CMPB    #CR,2(SP)               ;;IS CHARACTER A CARRIAGE RETURN?
13228  064552  001003                          BNE     1$                      ;;BRANCH IF NO
13229  064554  105037  064574                  CLRB    $CHARCNT                ;;YES--CLEAR CHARACTER COUNT
13230  064560  000406                          BR      $TYPEX                  ;;EXIT
13231  064562  122766  000012  000002  1$:     CMPB    #LF,2(SP)               ;;IS CHARACTER A LINE FEED?
13232  064570  001402                          BEQ     $TYPEX                  ;;BRANCH IF YES
13233  064572  105227                          INCB    (PC)+                   ;;COUNT THE CHARACTER
13234  064574  000000          $CHARCNT:.WORD  0                               ;;CHARACTER COUNT STORAGE
13235  064576  000207          $TYPEX: RTS     PC
13236
13237                          .SBTTL  SCOPE HANDLER ROUTINE
13238
13239                          ;;**********************************************************************
13240                          ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
13241                          ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
13242                          ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
13243                          ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
13244                          ;*SW14=1        LOOP ON TEST
13245                          ;*SW11=1        INHIBIT ITERATIONS
13246                          ;*SW09=1        LOOP ON ERROR
13247                          ;*SW08=1        LOOP ON TEST IN SWR<7:0>
13248                          ;*CALL
13249                          ;*      SCOPE           ;;SCOPE=IOT
13250
13251  064600                  $SCOPE:
13252  064600  104410                          CKSWR                           ;;TEST  FOR CHANGE IN SOFT-SWR
13253  064602  004737  063354                  JSR     PC,STOP
13254  064606  032777  040000  114340  1$:     BIT     #BIT14,@SWR             ;;LOOP ON PRESENT TEST?
13255  064614  001131                          BNE     $OVER                   ;;YES IF SW14=1
13256                          ;#####START OF CODE FOR THE XOR TESTER#####
13257  064616  000416          $XTSTR: BR      6$                              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
13258                                                                          ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
13259  064620  013746  000004                  MOV     @#ERRVEC,-(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
13260  064624  012737  064644  000004          MOV     #5$,@#ERRVEC            ;;SET FOR TIMEOUT
13261  064632  005737  177060                  TST     @#177060                ;;TIME OUT ON XOR?
13262  064636  012637  000004                  MOV     (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
13263  064642  000500                          BR      $SVLAD                  ;;GO TO THE NEXT TEST
13264  064644  022626          5$:     CMP     (SP)+,(SP)+                     ;;CLEAR THE STACK AFTER A TIME OUT
13265  064646  012637  000004                  MOV     (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
13266  064652  000440                          BR      7$                      ;;LOOP ON THE PRESENT TEST
13267  064654                  6$:;#####END OF CODE FOR THE XOR TESTER#####
13268  064654  032777  000400  114272          BIT     #BIT08,@SWR             ;;LOOP ON SPEC. TEST?
13269  064662  001421                          BEQ     2$                      ;;BR IF NO
13270  064664  005046                          CLR     -(SP)                   ;;CLEAR A TEMP. LOCATION
13271  064666  117716  114262                  MOVB    @SWR,(SP)               ;;PICKUP THE DESIRED TEST NUMBER
13272  064672  001414                          BEQ     8$                      ;;BRANCH IF BAD TEST NUMBER IN SWR
13273  064674  022716  000027                  CMP     #27,(SP)                ;;CHECK THE NUMBER IN THE SWR
13274  064700  002411                          BLT     8$                      ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
13275  064702  011637  001116                  MOV     (SP),$TSTNM             ;;UPDATE THE TEST NUMBER
13276  064706  005316                          DEC     (SP)                    ;;BACKUP BY ONE
13277  064710  006316                          ASL     (SP)                    ;;SCALE THE TEST NUMBER AS AN INDEX
13278  064712  062716  065116                  ADD     #$SW08TBL,(SP)          ;;FORM THE ADDRESS OF TEST POINTER
```

```
13279  064716  013637  001122                  MOV     @(SP)+,$LPADR   ;;SET LOOP ADDRESS TO DESIRED TEST
13280  064722  000466                          BR      $OVER           ;;GO LOOP ON THE TEST
13281  064724  005726                  8$:     TST     (SP)+           ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
13282  064726  105737  001117          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
13283  064732  001421                          BEQ     3$              ;;BR IF NO
13284  064734  123737  001131  001117          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
13285  064742  101015                          BHI     3$              ;;BR IF NO
13286  064744  032777  001000  114202          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
13287  064752  001404                          BEQ     4$              ;;BR IF NO
13288  064754  013737  001124  001122  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
13289  064762  000446                          BR      $OVER
13290  064764  105037  001117          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
13291  064770  005037  001206                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
13292  064774  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
13293  064776  032777  004000  114150  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
13294  065004  001011                          BNE     1$              ;;BR IF YES
13295  065006  005737  001230                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
13296  065012  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
13297  065014  005237  001120                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
13298  065020  023737  001206  001120          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
13299  065026  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
13300  065030  012737  000001  001120  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
13301  065036  013737  065114  001206          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
13302  065044  105237  001116          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
13303  065050  113737  001116  001226          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
13304  065056  011637  001122                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
13305  065062  011637  001124                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
13306  065066  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
13307  065072  112737  000001  001131          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
13308  065100  013777  001116  114050  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
13309  065106  013716  001122                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
13310  065112  000002                          RTI                     ;;FIXES PS
13311  065114  000012          $MXCNT: 10.                             ;;MAX. NUMBER OF ITERATIONS
13312  065116                  $SW08TBL:
13313  065116  007140                          .WORD   TST1+2          ;;STARTING ADDRESS OF TEST 1
13314  065120  007336                          .WORD   TST2+2          ;;STARTING ADDRESS OF TEST 2
13315  065122  007522                          .WORD   TST3+2          ;;STARTING ADDRESS OF TEST 3
13316  065124  007704                          .WORD   TST4+2          ;;STARTING ADDRESS OF TEST 4
13317  065126  010726                          .WORD   TST5+2          ;;STARTING ADDRESS OF TEST 5
13318  065130  011720                          .WORD   TST6+2          ;;STARTING ADDRESS OF TEST 6
13319  065132  013160                          .WORD   TST7+2          ;;STARTING ADDRESS OF TEST 7
13320  065134  014202                          .WORD   TST10+2         ;;STARTING ADDRESS OF TEST 10
13321  065136  015174                          .WORD   TST11+2         ;;STARTING ADDRESS OF TEST 11
13322  065140  016436                          .WORD   TST12+2         ;;STARTING ADDRESS OF TEST 12
13323  065142  017456                          .WORD   TST13+2         ;;STARTING ADDRESS OF TEST 13
13324  065144  021004                          .WORD   TST14+2         ;;STARTING ADDRESS OF TEST 14
13325  065146  022024                          .WORD   TST15+2         ;;STARTING ADDRESS OF TEST 15
13326  065150  023044                          .WORD   TST16+2         ;;STARTING ADDRESS OF TEST 16
13327  065152  024742                          .WORD   TST17+2         ;;STARTING ADDRESS OF TEST 17
13328  065154  026416                          .WORD   TST20+2         ;;STARTING ADDRESS OF TEST 20
13329  065156  027524                          .WORD   TST21+2         ;;STARTING ADDRESS OF TEST 21
13330  065160  030472                          .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
13331  065162  031550                          .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
13332  065164  032636                          .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
13333  065166  033630                          .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
13334  065170  034652                          .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
```

```
13335  065172  035674                          .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
13336                                  .SBTTL  ERROR HANDLER ROUTINE
13337
13338                                  ;;**********************************************************************
13339                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
13340                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
13341                                  ;*AND GO TO ERRTYP ON ERROR
13342                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
13343                                  ;*SW15=1         HALT ON ERROR
13344                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
13345                                  ;*SW10=1         BELL ON ERROR
13346                                  ;*SW09=1         LOOP ON ERROR
13347                                  ;*CALL
13348                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
13349
13350  065174                          $ERROR:
13351  065174  104410                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
13352  065176  105237  001117          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
13353  065202  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
13354  065204  013777  001116  113744          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
13355  065212  032777  002000  113734          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
13356  065220  001402                          BEQ     1$              ;;NO - SKIP
13357  065222  104401  001212                  TYPE    ,$BELL          ;;RING BELL
13358  065226  005237  001126          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
13359  065232  011637  001132                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
13360  065236  162737  000002  001132          SUB     #2,$ERRPC
13361  065244  117737  113662  001130          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
13362  065252  032777  020000  113674          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
13363  065260  001004                          BNE     20$             ;;SKIP TYPEOUTS
13364  065262  004737  037216                  JSR     PC,ERRTYP       ;;GO TO USER ERROR ROUTINE
13365  065266  104401  001217                  TYPE    ,$CRLF
13366  065272                          20$:
13367  065272  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
13368  065300  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
13369  065302  113737  001130  065314          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
13370  065310  004737  067236                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
13371  065314     000          21$:    .BYTE   0
13372  065315     000                  .BYTE   0
13373  065316  000777          22$:    BR      22$             ;;APT ERROR LOOP
13374  065320  005777  113630  2$:     TST     @SWR            ;;HALT ON ERROR
13375  065324  100002                  BPL     3$              ;;SKIP IF CONTINUE
13376  065326  000000                  HALT                    ;;HALT ON ERROR!
13377  065330  104410                  CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
13378  065332  032777  001000  113614  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
13379  065340  001402                  BEQ     4$              ;;BR IF NO
13380  065342  013716  001124                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
13381  065346  005737  001210          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
13382  065352  001402                  BEQ     5$              ;;BR IF NONE
13383  065354  013716  001210                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
13384  065360                          5$:
13385  065360  022737  037176  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
13386  065366  001001                  BNE     6$              ;;BRANCH IF NO
13387  065370  000000                  HALT                    ;;YES
13388  065372                          6$:
13389  065372  000002                  RTI                     ;;RETURN
13390                                  .SBTTL  TTY INPUT ROUTINE
```

```
13391
13392                                   ;;*****************************************************************
13393                                   .ENABL  LSB
13394  065374  000000                   $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
13395  065376  000000                   $TKQIN: .WORD   0               ;;INPUT POINTER
13396  065400  000000                   $TKQOUT: .WORD  0               ;;OUTPUT POINTER
13397  065402  000001                   $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
13398          065403                   $TKQEND=.
13399          065404                   .EVEN
13400
13401                                   ;*TK INITIALIZE ROUTINE
13402                                   ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
13403                                   ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
13404                                   ;
13405                                   ;*CALL:
13406                                   ;*      JSR     PC,$TKINT
13407                                   ;*      RETURN
13408                                   ;
13409  065404  005037  065374           $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
13410  065410  012737  065402  065376           MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
13411  065416  013737  065376  065400           MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
13412  065424  012737  065454  000060           MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
13413  065432  012737  000200  000062           MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
13414  065440  005777  113516                   TST     @$TKB           ;;CLEAR DONE FLAG
13415  065444  012777  000100  113506           MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
13416  065452  000207                           RTS     PC              ;;RETURN TO CALLER
13417
13418                                   ;*TK SERVICE ROUTINE
13419                                   ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
13420                                   ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
13421                                   ;*IT IN THE QUEUE.
13422                                   ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
13423                                   ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (SHUT2)
13424                                   ;
13425  065454  117746  113502           $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
13426  065460  042716  177600                   BIC     #↑C177,(SP)     ;;STRIP THE JUNK
13427  065464  021627  000003                   CMP     (SP),#3         ;;IS IT A CONTROL C?
13428  065470  001007                           BNE     1$              ;;BRANCH IF NO
13429  065472  104401  066570                   TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (↑C)
13430  065476  004737  065404                   JSR     PC,$TKINT       ;;INIT THE KEYBOARD
13431  065502  005726                           TST     (SP)+           ;;CLEAN UP STACK
13432  065504  000137  063464                   JMP     SHUT2           ;;CONTROL C RESTART
13433  065510  021627  000007           1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
13434  065514  001004                           BNE     2$              ;;BRANCH IF NO
13435  065516  022737  000176  001154           CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
13436  065524  001500                           BEQ     6$              ;;GO TO SWR CHANGE
13437
13438  065526                           2$:
13439  065526  022737  000001  065374           CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
13440  065534  001004                           BNE     3$              ;;BRANCH IF NO
13441  065536  104401  001212                   TYPE    ,$BELL          ;;RING THE TTY BELL
13442  065542  005726                           TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
13443  065544  000451                           BR      5$              ;;EXIT
13444  065546  021627  000023           3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
13445  065552  001021                           BNE     32$             ;;BRANCH IF NO
13446  065554  005077  113400                   CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
```

```
13447  065560  005726                  TST     (SP)+           ;;CLEAN CHAR OFF STACK
13448  065562  105777  113372  31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
13449  065566  100375                  BPL     31$             ;;LOOP UNTIL ITS THERE
13450  065570  117746  113366          MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
13451  065574  042716  177600          BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
13452  065600  022627  000021          CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
13453  065604  001366                  BNE     31$             ;;BRANCH IF NO
13454  065606  012777  000100  113344  MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
13455  065614  000002                  RTI                     ;;RETURN
13456  065616  005237  065374  32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
13457  065622  021627  000140          CMP     (SP),#140       ;;IS IT UPPER CASE?
13458  065626  002405                  BLT     4$              ;;BRANCH IF YES
13459  065630  021627  000175          CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
13460  065634  003002                  BGT     4$              ;;BRANCH IF YES
13461  065636  042716  000040          BIC     #40,(SP)        ;;MAKE IT UPPER CASE
13462  065642  112677  177530  4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
13463  065646  005237  065376          INC     $TKQIN          ;;UPDATE THE POINTER
13464  065652  023727  065376  065403  CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
13465  065660  001003                  BNE     5$              ;;BRANCH IF NO
13466  065662  012737  065402  065376  MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
13467  065670  000002          5$:     RTI                     ;;RETURN
13468
13469                          ;;**********************************************************************
13470                          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
13471                          ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
13472                          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
13473                          ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
13474  065672  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
13475  065700  001124                  BNE     15$             ;;EXIT IF NOT
13476  065702  105777  113252          TSTB    @$TKS           ;;IS A CHAR WAITING?
13477  065706  100121                  BPL     15$             ;;IF NOT, EXIT
13478  065710  117746  113246          MOVB    @$TKB,-(SP)     ;;YES
13479  065714  042716  177600          BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
13480  065720  021627  000007          CMP     (SP),#7         ;;IS IT A CONTROL-G?
13481  065724  001300                  BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
13482                                                          ;;AND EXIT
13483
13484                          ;;**********************************************************************
13485                          ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
13486                          ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
13487                          ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
13488  065726  123727  001150  000001  6$:      CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
13489  065734  001674                  BEQ     2$              ;;BRANCH IF YES
13490  065736  005726                  TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
13491  065740  004737  065404          JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
13492  065744  005077  113210          CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
13493  065750  112737  000001  001151  MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
13494
13495  065756  104401  066602          TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
13496  065762  104401  066607  $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
13497  065766  013746  000176          MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
13498  065772  104402                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
13499  065774  104401  066620          TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
13500  066000  005046          19$:    CLR     -(SP)           ;;CLEAR COUNTER
13501  066002  005046                  CLR     -(SP)           ;;THE NEW SWR
13502  066004  105777  113150  7$:     TSTB    @$TKS           ;;CHAR THERE?
```

```
13503  066010  100375                        BPL     7$              ;;IF NOT TRY AGAIN
13504
13505  066012  117746  113144                MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
13506  066016  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
13507
13508  066022  021627  000003                CMP     (SP),#3         ;;IS IT A CONTROL-C?
13509  066026  001015                        BNE     9$              ;;BRANCH IF NOT
13510  066030  104401  066570                TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
13511  066034  062706  000006                ADD     #6,SP           ;;CLEAN UP STACK
13512  066040  123727  001151  000001        CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
13513  066046  001003                        BNE     8$              ;;BRANCH IF NO
13514  066050  012777  000100  113102        MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
13515  066056  000137  063464        8$:     JMP     SHUT2           ;;CONTROL-C RESTART
13516
13517
13518  066062  021627  000025        9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
13519  066066  001005                        BNE     10$             ;;BRANCH IF NOT
13520  066070  104401  066575                TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
13521  066074  062706  000006        20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
13522  066100  000737                        BR      19$             ;;LET'S TRY IT AGAIN
13523
13524
13525  066102  021627  000015        10$:    CMP     (SP),#15        ;;IS IT A <CR>?
13526  066106  001022                        BNE     16$             ;;BRANCH IF NO
13527  066110  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
13528  066114  001403                        BEQ     11$             ;;BRANCH IF YES
13529  066116  016677  000002  113030        MOV     2(SP),@SWR      ;;SAVE NEW SWR
13530  066124  062706  000006        11$:    ADD     #6,SP           ;;CLEAR UP STACK
13531  066130  104401  001217        14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
13532  066134  123727  001151  000001        CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
13533  066142  001003                        BNE     15$             ;;BRANCH IF NOT
13534  066144  012777  000100  113006        MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
13535  066152  000002        15$:    RTI                             ;;RETURN
13536  066154  004737  064530        16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
13537  066160  021627  000060                CMP     (SP),#60        ;;CHAR < 0?
13538  066164  002420                        BLT     18$             ;;BRANCH IF YES
13539  066166  021627  000067                CMP     (SP),#67        ;;CHAR > 7?
13540  066172  003015                        BGT     18$             ;;BRANCH IF YES
13541  066174  042726  000060                BIC     #60,(SP)+       ;;STRIP-OFF ASCII
13542  066200  005766  000002                TST     2(SP)           ;;IS THIS THE FIRST CHAR
13543  066204  001403                        BEQ     17$             ;;BRANCH IF YES
13544  066206  006316                        ASL     (SP)            ;;NO, SHIFT PRESENT
13545  066210  006316                        ASL     (SP)            ;;    CHAR OVER TO MAKE
13546  066212  006316                        ASL     (SP)            ;;    ROOM FOR NEW ONE.
13547  066214  005266  000002        17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
13548  066220  056616  177776                BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
13549  066224  000667                        BR      7$              ;;GET THE NEXT ONE
13550  066226  104401  001216        18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
13551  066232  000720                        BR      20$             ;;SIMULATE CONTROL-U
13552                                .DSABL  LSB
13553
13554
13555                                ;;*****************************************************************
13556                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
13557                                ;*CALL:
13558                                ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
```

```
13559                                   ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
13560                                   ;*                              ;;WITH PARITY BIT STRIPPED OFF
13561                                   ;
13562
13563  066234  011646                   $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
13564  066236  016666  000004  000002           MOV     4(SP),2(SP)     ;;THE PS
13565  066244  005066  000004                   CLR     4(SP)           ;;GET READY FOR A CHARACTER
13566  066250  005046                           CLR     -(SP)           ;;PUT NEW PS ON STACK
13567  066252  012746  066260                   MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
13568  066256  000002                           RTI                     ;;POP NEW PC AND PS
13569  066260                          64$:
13570  066260  005737  065374           1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
13571  066264  001775                           BEQ     1$
13572  066266  005337  065374                   DEC     $TKCNT          ;;DECREMENT THE COUNTER
13573  066272  117766  177102  000004           MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
13574  066300  005237  065400                   INC     $TKQOUT         ;;UPDATE THE POINTER
13575  066304  023727  065400  065403           CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
13576  066312  001003                           BNE     2$              ;;BRANCH IF NO
13577  066314  012737  065402  065400           MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
13578  066322  000002           2$:             RTI                     ;;RETURN
13579                                  ;;*************************************************************
13580                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
13581                                  ;*CALL:
13582                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
13583                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
13584                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
13585
13586  066324  010346                   $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
13587  066326  005046                           CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
13588  066330  012703  066560           1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
13589  066334  022703  066570           2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
13590  066340  101456                           BLOS    4$              ;;BR IF YES
13591  066342  104411                           RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
13592  066344  112613                           MOVB    (SP)+,(R3)      ;;GET CHARACTER
13593  066346  122713  000177           10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
13594  066352  001022                           BNE     5$              ;;BR IF NO
13595  066354  005716                           TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
13596  066356  001007                           BNE     6$              ;;BR IF NO
13597  066360  112737  000134  066556           MOVB    #'\,9$          ;;TYPE A BACK SLASH
13598  066366  104401  066556                   TYPE    9$
13599  066372  012716  177777                   MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
13600  066376  005303           6$:             DEC     R3              ;;BACKUP BY ONE
13601  066400  020327  066560                   CMP     R3,#$TTYIN      ;;STACK EMPTY?
13602  066404  103434                           BLO     4$              ;;BR IF YES
13603  066406  111337  066556                   MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
13604  066412  104401  066556                   TYPE    9$              ;;GO TYPE
13605  066416  000746                           BR      2$              ;;GO READ ANOTHER CHAR.
13606  066420  005716           5$:             TST     (SP)            ;;RUBOUT KEY SET?
13607  066422  001406                           BEQ     7$              ;;BR IF NO
13608  066424  112737  000134  066556           MOVB    #'\,9$          ;;TYPE A BACK SLASH
13609  066432  104401  066556                   TYPE    9$
13610  066436  005016                           CLR     (SP)            ;;CLEAR THE RUBOUT KEY
13611  066440  122713  000025           7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
13612  066444  001003                           BNE     8$              ;;BR IF NO
13613  066446  104401  066575                   TYPE    $CNTLU          ;;TYPE A CONTROL "U"
13614  066452  000726                           BR      1$              ;;GO START OVER
```

```
13615  066454  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
13616  066460  001011                          BNE     3$              ;;BRANCH IF NO
13617  066462  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
13618  066464  104401  001217                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
13619  066470  104401  066560                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
13620  066474  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
13621  066476  104401  001216          4$:     TYPE    ,$QUES          ;;TYPE A '?'
13622  066502  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
13623  066504  111337  066556          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
13624  066510  104401  066556                  TYPE    ,9$             ;;
13625  066514  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
13626  066520  001305                          BNE     2$              ;;LOOP IF NOT RETURN
13627  066522  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
13628  066526  104401  001220                  TYPE    ,$LF            ;;TYPE A LINE FEED
13629  066532  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
13630  066534  012603                          MOV     (SP)+,R3        ;;RESTORE R3
13631  066536  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
13632  066540  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
13633  066546  012766  066560  000004          MOV     #$TTYIN,4(SP)   ;;
13634  066554  000002                          RTI                     ;;RETURN
13635  066556     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
13636  066557     000                  .BYTE   0                       ;;TERMINATOR
13637  066560  000010          $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
13638  066570  041536  005015    000   $CNTLC: .ASCIZ  /↑C/<15><12>    ;;CONTROL "C"
13639  066575     136  006525  000012  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
13640  066602  043536  005015    000   $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
13641  066607     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
13642  066614  036440  000040
13643  066620  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
13644  066626  020075     000
13645          066632                  .EVEN
13646                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
13647
13648                                  ;;*****************************************************************
13649                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
13650                                  ;*CHANGE IT TO BINARY.
13651                                  ;*CALL:
13652                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
13653                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
13654                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
13655
13656  066632  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
13657  066634  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
13658  066642  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
13659  066644  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
13660  066646  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
13661  066650  104412                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
13662  066652  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
13663  066654  005001                          CLR     R1              ;;CLEAR DATA WORD
13664  066656  005002                          CLR     R2
13665  066660  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
13666  066662  001412                          BEQ     3$              ;;IF ZERO GET OUT
13667  066664  006301                          ASL     R1              ;;*2
13668  066666  006102                          ROL     R2
13669  066670  006301                          ASL     R1              ;;*4
13670  066672  006102                          ROL     R2
```

```
13671  066674  006301                          ASL     R1              ;;*8
13672  066676  006102                          ROL     R2
13673  066700  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
13674  066704  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
13675  066706  000764                          BR      2$              ;;LOOP
13676  066710  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
13677  066712  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
13678  066716  010237  066732                  MOV     R2,$HIOCT
13679  066722  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
13680  066724  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
13681  066726  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
13682  066730  000002                          RTI                     ;;RETURN
13683  066732  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
13684                                  .SBTTL  TRAP DECODER
13685
13686                                  ;;**************************************************************
13687                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
13688                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
13689                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
13690                                  ;*GO TO THAT ROUTINE.
13691
13692  066734  016646  000002  .       $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
13693  066740  042716  000020                  BIC     #20,(SP)        ;;  THE CALLER--DO NOT ALLOW
13694  066744  012746  066752                  MOV     #1$,-(SP)       ;;  T-BIT TRAPS
13695  066750  000002                          RTI                     ;;SET THE NEW STATUS
13696  066752  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
13697  066754  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
13698  066760  005740                          TST     -(R0)           ;;BACKUP BY 2
13699  066762  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
13700  066764  006300                          ASL     R0              ;;POSITION FOR INDEXING
13701  066766  016000  067006                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
13702  066772  000200                          RTS     R0              ;;GO TO ROUTINE
13703
13704
13705                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
13706
13707  066774  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
13708  066776  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
13709  067004  000002                          RTI                     ;;RESTORE THE PSW
13710
13711                                  .SBTTL  TRAP TABLE
13712
13713                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
13714                                  ;*BY THE "TRAP" INSTRUCTION.
13715
13716                                  ;       ROUTINE
13717                                  ;       -------
13718  067006  066774                  $TRPAD: .WORD   $TRAP2
13719  067010  064316                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
13720  067012  064114                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
13721  067014  064070                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
13722  067016  064130                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
13723  067020  063644                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
13724  067022  063570                          $TYPBN  ;;CALL=TYPBN    TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER
13725
13726  067024  065762                          $GTSWR  ;;CALL=GTSWR    TRAP+7(104407)  GET SOFT-SWR SETTING
```

```
13727
13728  067026  065672                          $CKSWR  ;;CALL=CKSWR    TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
13729  067030  066234                          $RDCHR  ;;CALL=RDCHR    TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
13730  067032  066324                          $RDLIN  ;;CALL=RDLIN    TRAP+12(104412) TTY TYPEIN STRING ROUTINE
13731  067034  066632                          $RDOCT  ;;CALL=RDOCT    TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
13732  067036  063474                          $SAVREG ;;CALL=SAVREG   TRAP+14(104414) SAVE R0-R5 ROUTINE
13733  067040  063532                          $RESREG ;;CALL=RESREG   TRAP+15(104415) RESTORE R0-R5 ROUTINE
13734                                  .SBTTL  POWER DOWN AND UP ROUTINES
13735
13736                                  ;;**********************************************************************
13737                                  ;POWER DOWN ROUTINE
13738  067042  012737  067202  000024 $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
13739  067050  012737  000340  000026         MOV     #340,@#PWRVEC+2  ;;PRIO:7
13740  067056  010046                         MOV     R0,-(SP)         ;;PUSH R0 ON STACK
13741  067060  010146                         MOV     R1,-(SP)         ;;PUSH R1 ON STACK
13742  067062  010246                         MOV     R2,-(SP)         ;;PUSH R2 ON STACK
13743  067064  010346                         MOV     R3,-(SP)         ;;PUSH R3 ON STACK
13744  067066  010446                         MOV     R4,-(SP)         ;;PUSH R4 ON STACK
13745  067070  010546                         MOV     R5,-(SP)         ;;PUSH R5 ON STACK
13746  067072  017746  112056                 MOV     @$WR,-(SP)       ;;PUSH @$WR ON STACK
13747  067076  010637  067206                 MOV     SP,$SAVR6        ;;SAVE SP
13748  067102  012737  067114  000024         MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
13749  067110  000000                         HALT
13750  067112  000776                         BR      .-2              ;;HANG UP
13751
13752                                  ;;**********************************************************************
13753                                  ;POWER UP ROUTINE
13754  067114  012737  067202  000024 $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
13755  067122  013706  067206                 MOV     $SAVR6,SP        ;;GET SP
13756  067126  005037  067206                 CLR     $SAVR6           ;;WAIT LOOP FOR THE TTY
13757  067132  005237  067206         1$:     INC     $SAVR6           ;;WAIT FOR THE INC
13758  067136  001375                         BNE     1$               ;;OF  WORD
13759  067140  012677  112010                 MOV     (SP)+,@$WR       ;;POP STACK INTO @$WR
13760  067144  012605                         MOV     (SP)+,R5         ;;POP STACK INTO R5
13761  067146  012604                         MOV     (SP)+,R4         ;;POP STACK INTO R4
13762  067150  012603                         MOV     (SP)+,R3         ;;POP STACK INTO R3
13763  067152  012602                         MOV     (SP)+,R2         ;;POP STACK INTO R2
13764  067154  012601                         MOV     (SP)+,R1         ;;POP STACK INTO R1
13765  067156  012600                         MOV     (SP)+,R0         ;;POP STACK INTO R0
13766  067160  012737  067042  000024         MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
13767  067166  012737  000340  000026         MOV     #340,@#PWRVEC+2  ;;PRIO:7
13768  067174  104401                         TYPE                     ;REPORT THE POWER FAILURE
13769  067176  067210                 $PWRMG: .WORD   $POWER           ;;POWER FAIL MESSAGE POINTER
13770  067200  000002                         RTI
13771  067202  000000                 $ILLUP: HALT                     ;;THE POWER UP SEQUENCE WAS STARTED
13772  067204  000776                         BR      .-2              ;;  BEFORE THE POWER DOWN WAS COMPLETE
13773  067206  000000                 $SAVR6: 0                        ;;PUT THE SP HERE
13774  067210  005015  047520  042527 $POWER: .ASCIZ  <15><12>"POWER"
13775  067216  000122
13776                                          .EVEN
13777                                  .SBTTL  APT COMMUNICATIONS ROUTINE
13778
13779                                  ;;**********************************************************************
13780  067220  112737  000001  067464 $ATY1:  MOVB    #1,$FFLG         ;;TO REPORT FATAL ERROR
13781  067226  112737  000001  067462 $ATY3:  MOVB    #1,$MFLG         ;;TO TYPE A MESSAGE
13782  067234  000403                         BR      $ATYC
```

```
13783  067236  112737  000001  067464  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
13784  067244                  $ATYC:
13785  067244  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
13786  067246  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
13787  067250  105737  067462          TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
13788  067254  001450                  BEQ     5$              ;;IF NOT:  BR
13789  067256  122737  000001  001242  CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
13790  067264  001031                  BNE     3$              ;;IF NOT:  BR
13791  067266  132737  000100  001243  BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
13792  067274  001425                  BEQ     3$              ;;IF NOT:  BR
13793  067276  017600  000004          MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
13794  067302  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
13795  067310  005737  001222  1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
13796  067314  001375                  BNE     1$              ;;IF NOT:  WAIT
13797  067316  010037  001236          MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
13798  067322  105720          2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
13799  067324  001376                  BNE     2$
13800  067326  163700  001236          SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
13801  067332  006200                  ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
13802  067334  010037  001240          MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
13803  067340  012737  000004  001222  MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
13804  067346  000413                  BR      5$
13805  067350  017637  000004  067374  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
13806  067356  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
13807  067364  013746  177776          MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
13808  067370  004737  064316          JSR     PC,$TYPE        ;;CALL TYPE MACRO
13809  067374  000000          4$:     .WORD   0
13810  067376                  5$:
13811  067376  105737  067464  10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
13812  067402  001416                  BEQ     12$             ;;IF NOT:  BR
13813  067404  005737  001242          TST     $ENV            ;;RUNNING UNDER APT?
13814  067410  001413                  BEQ     12$             ;;IF NOT:  BR
13815  067412  005737  001222  11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
13816  067416  001375                  BNE     11$             ;;IF NOT:  WAIT
13817  067420  017637  000004  001224  MOV     @4(SP),$FATAL   ;;GET ERROR #
13818  067426  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
13819  067434  005237  001222          INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
13820  067440  105037  067464  12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
13821  067444  105037  067463          CLRB    $LFLG           ;;CLEAR LOG FLAG
13822  067450  105037  067462          CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
13823  067454  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
13824  067456  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
13825  067460  000207                  RTS     PC              ;;RETURN
13826  067462     000  $MFLG:  .BYTE  0       ;;MESSG. FLAG
13827  067463     000  $LFLG:  .BYTE  0       ;;LOG FLAG
13828  067464     000  $FFLG:  .BYTE  0       ;;FATAL FLAG
13829          067466          .EVEN
13830          000200  APTSIZE=200
13831          000001  APTENV=001
13832          000100  APTSPOOL=100
13833          000040  APTCSUP=040
13834
13835                  .NLIST  BEX
```

```
                                .SBTTL  CONSOLE MESSAGES

067466                          SCTMSG:
067466  005015  040503  047116  .ASCII  <CR><LF>@CANNOT RECOVER THE BAD SECTOR FILES FROM LAST @<CR><LF>
067550  051124  041501  020113  .ASCIZ  @TRACK FOR THIS DEVICE@
067576     051                  CLSPRN: .ASCII  @)@
067577     075     000          EQUALS: .ASCIZ  @=@
067601     015  025012     000  PROMPT: .ASCIZ  <CR><LF>@*@
067605     077     000          QSTMRK: .ASCIZ  @?@
067607                          HELPQST:
067607     015  006412  052012  .ASCII  <CR><LF><CR><LF>/THIS PROGRAM SHOULD BE HALTED BY TYPE ↑C./
067664  005015  005015  044124  .ASCII  <CR><LF><CR><LF>/THE PROGRAM WILL BE HALTED AT END OF PASS/
067741     015  052012  050131  .ASCIZ  <CR><LF>@TYPE HELP TEXT (Y OR N)??@
067775                          UBUSQST:
067775     015  041412  040510  .ASCIZ  <CR><LF>@CHANGE RM03 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N)<CR> ??@
070074  005015  051525  020105  CNSL00: .ASCIZ  <CR><LF>@USE SAME DEVICES (Y OR N) ??@
070133     015  051012  030115  CNSL01: .ASCIZ  <CR><LF>@RM03 BUS ADDRESS (@
070160  005015  047105  051124  CNSL02: .ASCII  <CR><LF>@ENTRY NOT IN I/O PAGE@
070207     015  040412  042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE >160000@
070241     015  051012  030115  CNSL03: .ASCIZ  <CR><LF>@RM03 VECTOR ADDRESS (@
070271     015  042412  052116  CNSL04: .ASCII  <CR><LF>@ENTRY OUT OF RANGE@
070315     015  040412  042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE <1000@
070345     015  051012  030115  CNSL05: .ASCIZ  <CR><LF>@RM03 INTERRUPT PRIORITY (@
070401     015  042412  052116  CNSL06: .ASCIZ  <CR><LF>@ENTRY OUT OF RANGE@
070426                          CNSL07:
070426  005015  054524  042520  .ASCII  <CR><LF>@TYPE (A) TO TEST ALL DEVICES, OR TYPE DEVICE@
070504  047040  046525  042502  .ASCII  @ NUMBER(S)@
070516  005015  042524  046522  .ASCIZ  <CR><LF>@TERMINATE INPUT WITH CARRIAGE RETURN@
070565     015  041412  040510  XDPMG:  .ASCII  <CR><LF>/CHANGE XXDP PACK,CLEAR LOC 40/
070624  005015  042522  052123          .ASCIZ  <CR><LF>/RESTART THE PROGRAM/
070652  005015  047516  020124  NOTEX:  .ASCIZ  <CR><LF>/NOT EXIST DRIVE  /

                                .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.

070676                  FNCDTB:                                 ;FUNCTION CODE TABLE
```

```
070676  020000          .WORD   OPI                                     ;NOP
070700  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (2)
070702  132000          .WORD   ATA!OPI!IVC!IAE                         ;SEEK
070704  130000          .WORD   ATA!OPI!IVC                             ;RECALIBRATE
070706  020000          .WORD   OPI                                     ;DRIVE CLEAR
070710  030000          .WORD   OPI!IVC                                 ;RELEASE
070712  130000          .WORD   OPI!ATA!IVC                             ;OFFSET
070714  130000          .WORD   OPI!ATA!IVC                             ;RETURN TO CENTERLINE
070716  020000          .WORD   OPI                                     ;READ IN PRESET
070720  020000          .WORD   OPI                                     ;PACK ACKNOWLEDGE
070722  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (24)
070724  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (26)
070726  132000          .WORD   ATA!OPI!IVC!IAE                         ;SEARCH
070730  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (32)
070732  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (34)
070734  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (36)
070736  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (40)
070740  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (42)
070742  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (44)
070744  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (46)
070746  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK DATA
070750  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK HEADER AND DATA
070752  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (54)
070754  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (56)
070756  037200          .WORD   OPI!IVC!WLE!IAE!AOE!HCE                 ;WRITE DATA
070760  037000          .WORD   OPI!IVC!WLE!IAE!AOE                     ;WRITE HEADER AND DATA
070762  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (64)
070764  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (66)
070766  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH                 ;READ DATA
070770  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH                 ;READ HEADER AND DATA
070772  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (74)
070774  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (76)
```

```
                        .SBTTL  ATTENTION (ATA) TABLE

070776  001             ATNTBL: .BYTE   1.
070777  002                     .BYTE   2.
071000  004                     .BYTE   4.
071001  010                     .BYTE   8.
071002  020                     .BYTE   16.
071003  040                     .BYTE   32.
071004  100                     .BYTE   64.
071005  200                     .BYTE   128.
```

```
                        .SBTTL  DATA PATTERN TABLE

071006                  RGDTPT:
071006                  MIXED:
071006  000000                  .WORD   0.
071010  000001                  .WORD   1.
071012  000003                  .WORD   3.
071014  000007                  .WORD   7.
071016  000017                  .WORD   15.
071020  000037                  .WORD   31.
071022  000077                  .WORD   63.
071024  000177                  .WORD   127.
071026  000377                  .WORD   255.
071030  000777                  .WORD   511.
071032  001777                  .WORD   1023.
071034  003777                  .WORD   2047.
071036  007777                  .WORD   4095.
071040  017777                  .WORD   8191.
071042  037777                  .WORD   16383.
071044  077777                  .WORD   32767.
071046  177777          ONES:   .WORD   65535.
071050  177777                  .WORD   65535.
071052  077777                  .WORD   32767.
071054  037777                  .WORD   16383.
071056  017777                  .WORD   8191.
071060  007777                  .WORD   4095.
071062  003777                  .WORD   2047.
071064  001777                  .WORD   1023.
071066  000777                  .WORD   511.
071070  000377                  .WORD   255.
071072  000177                  .WORD   127.
071074  000077                  .WORD   63.
071076  000037                  .WORD   31.
071100  000017                  .WORD   15.
071102  000007                  .WORD   7.
071104  000003                  .WORD   3.
071106  000001                  .WORD   1.
071110  000000          ZEROS:  .WORD   0.
071112  000000                  .WORD   0.
071114  000001                  .WORD   1.
071116  000002                  .WORD   2.
071120  000004                  .WORD   4.
071122  000010                  .WORD   8.
071124  000020                  .WORD   16.
071126  000040                  .WORD   32.
071130  000100                  .WORD   64.
071132  000200                  .WORD   128.
071134  000400                  .WORD   256.
071136  001000                  .WORD   512.
071140  002000                  .WORD   1024.
071142  004000                  .WORD   2048.
071144  010000                  .WORD   4096.
071146  020000                  .WORD   8192.
071150  040000                  .WORD   16384.
071152  100000                  .WORD   32768.
071154  100000                  .WORD   32768.
```

```
071156  040000                          .WORD   16384.
071160  020000                          .WORD   8192.
071162  010000                          .WORD   4096.
071164  004000                          .WORD   2048.
071166  002000                          .WORD   1024.
071170  001000                          .WORD   512.
071172  000400                          .WORD   256.
071174  000200                          .WORD   128.
071176  000100                          .WORD   64.
071200  000040                          .WORD   32.
071202  000020                          .WORD   16.
071204  000010                          .WORD   8.
071206  000004                          .WORD   4.
071210  000002                          .WORD   2.
071212  000001                          .WORD   1.
071214  000000                          .WORD   0.
071216  177777                          .WORD   65535.
071220  177776                          .WORD   65534.
071222  177774                          .WORD   65532.
071224  177770                          .WORD   65528.
071226  177760                          .WORD   65520.
071230  177740                          .WORD   65504.
071232  177700                          .WORD   65472.
071234  177600                          .WORD   65408.
071236  177400                          .WORD   65280.
071240  177000                          .WORD   65024.
071242  176000                          .WORD   64512.
071244  174000                          .WORD   63488.
071246  170000                          .WORD   61440.
071250  160000                          .WORD   57344.
071252  140000                          .WORD   49152.
071254  100000                          .WORD   32768.
071256  000000                          .WORD   0.
071260  000000                          .WORD   0.
071262  100000                          .WORD   32768.
071264  140000                          .WORD   49152.
071266  160000                          .WORD   57344.
071270  170000                          .WORD   61440.
071272  174000                          .WORD   63488.
071274  176000                          .WORD   64512.
071276  177000                          .WORD   65024.
071300  177400                          .WORD   65280.
071302  177600                          .WORD   65408.
071304  177700                          .WORD   65472.
071306  177740                          .WORD   65504.
071310  177760                          .WORD   65520.
071312  177770                          .WORD   65528.
071314  177774                          .WORD   65532.
071316  177776                          .WORD   65534.
071320  177777                          .WORD   65535.
071322  125252                  EARLY:  .WORD   43690.
071324  152525                          .WORD   43690./2
071326  125252                          .WORD   43690.
071330  177777                          .WORD   65535.
071332  177776                          .WORD   65534.
071334  177775                          .WORD   65533.
```

```
071336  177773                          .WORD   65531.
071340  177767                          .WORD   65527.
071342  177757                          .WORD   65519.
071344  177737                          .WORD   65503.
071346  177677                          .WORD   65471.
071350  177577                          .WORD   65407.
071352  177377                          .WORD   65279.
071354  176777                          .WORD   65023.
071356  175777                          .WORD   64511.
071360  173777                          .WORD   63487.
071362  167777                          .WORD   61439.
071364  157777                          .WORD   57343.
071366  137777                          .WORD   49151.
071370  077777                          .WORD   32767.
071372  077777                          .WORD   32767.
071374  137777                          .WORD   49151.
071376  157777                          .WORD   57343.
071400  167777                          .WORD   61439.
071402  173777                          .WORD   63487.
071404  175777                          .WORD   64511.
071406  176777                          .WORD   65023.
071410  177377                          .WORD   65279.
071412  177577                          .WORD   65407.
071414  177677                          .WORD   65471.
071416  177737                          .WORD   65503.
071420  177757                          .WORD   65519.
071422  177767                          .WORD   65527.
071424  177773                          .WORD   65531.
071426  177775                          .WORD   65533.
071430  177776                          .WORD   65534.
071432  177777                          .WORD   65535.
071434                          ENRGDT:
```

```
                                .SBTTL  ERROR MESSAGE TABLE

071434  076030  000000          EMT1:   .WORD   EMS1,0
071440  076077  076114  000000  EMT2:   .WORD   EMS2,EMS3,0
071446  076077  076157  000000  EMT3:   .WORD   EMS2,EMS4,0
071454  076222  076252  000000  EMT4:   .WORD   EMS5,EMS6,0
071462  076222  076364  000000  EMT5:   .WORD   EMS5,EMS10,0
071470  103057  100245  000000  EMT6:   .WORD   EMS167,EMS64,0
071476  101013  103104  000000  EMT7:   .WORD   EMS110,EMS170,0
071504  076317  000000          EMT10:  .WORD   EMS7,0
071510  076364  000000          EMT11:  .WORD   EMS10,0
071514  076426  076437  000000  EMT12:  .WORD   EMS11,EMS12,0
071522  076500  076511  076522  EMT13:  .WORD   EMS13,EMS14,EMS15,EMS16,0
071534  076574  100245  000000  EMT14:  .WORD   EMS17,EMS64,0
071542  076426  076657  000000  EMT15:  .WORD   EMS11,EMS21,0
071550  076426  076702  077031  EMT16:  .WORD   EMS11,EMS22,EMS27,0
071560  076426  076716  077042  EMT17:  .WORD   EMS11,EMS23,EMS30,0
071570  076426  076744  077042  EMT20:  .WORD   EMS11,EMS24,EMS30,0
071600  076426  076773  077031  EMT21:  .WORD   EMS11,EMS25,EMS27,0
071610  076426  077010  077031  EMT22:  .WORD   EMS11,EMS26,EMS27,0
071620  076426  077052  077042  EMT23:  .WORD   EMS11,EMS31,EMS30,0
071630  076426  077101  077042  EMT24:  .WORD   EMS11,EMS32,EMS30,0
071640  076426  077130  077042  EMT25:  .WORD   EMS11,EMS33,EMS30,0
071650  076426  077156  077042  EMT26:  .WORD   EMS11,EMS34,EMS30,0
071660  076426  077227  077042  EMT27:  .WORD   EMS11,EMS35,EMS30,0
071670  076426  077256  077042  EMT30:  .WORD   EMS11,EMS36,EMS30,0
071700  076426  077305  077042  EMT31:  .WORD   EMS11,EMS37,EMS30,0
071710  076426  077333  077042  EMT32:  .WORD   EMS11,EMS40,EMS30,0
071720  076426  077362  077042  EMT33:  .WORD   EMS11,EMS41,EMS30,0
071730  076426  077410  077042  EMT34:  .WORD   EMS11,EMS42,EMS30,0
071740  076426  077437  077042  EMT35:  .WORD   EMS11,EMS43,EMS30,0
071750  076426  077466  077042  EMT36:  .WORD   EMS11,EMS44,EMS30,0
071760  076426  077541  077042  EMT37:  .WORD   EMS11,EMS45,EMS30,0
071770  100331  076634  000000  EMT40:  .WORD   EMS66,EMS20,0
071776  100517  102225  100525  EMT41:  .WORD   EMS75,EMS141,EMS76,0
072006  102316  102326  100453  EMT42:  .WORD   EMS144,EMS145,EMS72,EMS76,0
072020  077633  077751  100525  EMT43:  .WORD   EMS47,EMS53,EMS76,0
072030  100556  077751  100525  EMT44:  .WORD   EMS77,EMS53,EMS76,0
072040  100604  077751  100525  EMT45:  .WORD   EMS100,EMS53,EMS76,0
072050  100632  077751  100525  EMT46:  .WORD   EMS101,EMS53,EMS76,0
072060  100263  100245  000000  EMT47:  .WORD   EMS65,EMS64,0
072066  076500  076522  100217  EMT50:  .WORD   EMS13,EMS15,EMS63,0
072076  076426  077604  077042  EMT51:  .WORD   EMS11,EMS46,EMS30,EMS67,0
072110  077633  077751  100401  EMT52:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS140,EMS141,0
072126  077633  077751  100401  EMT53:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS141,EMS164,0
072144  077662  077751  100401  EMT54:  .WORD   EMS50,EMS53,EMS67,0
072154  102253  102270  077751  EMT55:  .WORD   EMS142,EMS143,EMS53,EMS67,0
072166  077711  100453  100401  EMT56:  .WORD   EMS51,EMS72,EMS67,EMS115,EMS50,EMS70,0
072204  103057  100463  100401  EMT57:  .WORD   EMS167,EMS73,EMS67,0
072214  100034  100401  101213  EMT60:  .WORD   EMS56,EMS67,EMS115,EMS150,EMS152,EMS70,0
072232  100440  100034  100401  EMT61:  .WORD   EMS71,EMS56,EMS67,EMS115,EMS150,EMS152,EMS72,0
072252  102204  100401  101213  EMT62:  .WORD   EMS140,EMS67,EMS115,EMS47,EMS70,0
072266  000000                  EMT63:  .WORD
072270  102411  102454  100426  EMT64:  .WORD   EMS150,EMS152,EMS70,EMS115,EMS56,EMS73,EMS67,0
072310  077737  103504  103665  EMT65:  .WORD   EMS52,EMS205,EMS214,EMS206,EMS115,EMS51,EMS72,0
072330  103016  100707  100453  EMT66:  .WORD   EMS165,EMS103,EMS72,EMS124,0
```

```
072342  103016  100707  100453  EMT67:  .WORD   EMS165,EMS103,EMS72,EMS171,0
072354  077604  076634  102711  EMT70:  .WORD   EMS46,EMS20,EMS163,0
072364  100440  100632  102711  EMT71:  .WORD   EMS71,EMS101,EMS163,0
072374  077633  100453  102711  EMT72:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS140,EMS141,0
072412  077633  100453  102711  EMT73:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS141,EMS72,0
072430  100034  077751  102711  EMT74:  .WORD   EMS56,EMS53,EMS163,0
072440  100440  100034  102711  EMT75:  .WORD   EMS71,EMS56,EMS163,EMS115,EMS150,EMS152,EMS72,0
072460  077662  077751  102711  EMT76:  .WORD   EMS50,EMS53,EMS163,0
072470  102253  102270  077751  EMT77:  .WORD   EMS142,EMS143,EMS53,EMS163,0
072502  100717  102225  102711  EMT100: .WORD   EMS75,EMS141,EMS163,EMS115,EMS47,EMS70,0
072520  100717  102411  102711  EMT101: .WORD   EMS75,EMS150,EMS163,EMS115,EMS56,EMS73,0
072536  103057  100463  102711  EMT102: .WORD   EMS167,EMS73,EMS163,0
072546  102374  102430  100500  EMT103: .WORD   EMS147,EMS151,EMS74,EMS163,0
072560  077711  100500  102711  EMT104: .WORD   EMS51,EMS74,EMS163,EMS115,EMS50,EMS70,0
072576  103016  100604  102711  EMT105: .WORD   EMS165,EMS100,EMS163,0
072606  103016  100556  102711  EMT106: .WORD   EMS165,EMS77,EMS163,0
072616  102316  102326  077751  EMT107: .WORD   EMS144,EMS145,EMS53,EMS163,EMS115,EMS143,EMS70,0
072636  101013  101053  101220  EMT110: .WORD   EMS110,EMS112,EMS116,EMS111,0
072650  101137  076157  000000  EMT111: .WORD   EMS113,EMS4,0
072656  101137  076114  000000  EMT112: .WORD   EMS113,EMS3,0
072664  101013  101213  101220  EMT113: .WORD   EMS110,EMS115,EMS116,EMS117,EMS114,0
072700  101137  101272  000000  EMT114: .WORD   EMS113,EMS120,0
072706  101307  101747  101773  EMT115: .WORD   EMS121,EMS132,EMS133,0
072716  101352  101747  101773  EMT116: .WORD   EMS122,EMS132,EMS133,0
072726  101407  101747  101773  EMT117: .WORD   EMS123,EMS132,EMS133,0
072736  101452  101747  101773  EMT120: .WORD   EMS124,EMS132,EMS133,
072746  101504  101747  101773  EMT121: .WORD   EMS125,EMS132,EMS133,0
072756  101547  101747  101773  EMT122: .WORD   EMS126,EMS132,EMS133,0
072766  102147  101747  101773  EMT123: .WORD   EMS137,EMS132,EMS133,0
072776  101651  101747  101773  EMT124: .WORD   EMS130,EMS132,EMS133,0
073006  101707  101747  101773  EMT125: .WORD   EMS131,EMS132,EMS133,0
073016  101307  101747  102016  EMT126: .WORD   EMS121,EMS132,EMS134,EMS123,0
073030  101352  101747  102016  EMT127: .WORD   EMS122,EMS132,EMS134,EMS123,0
073042  101407  101747  102016  EMT130: .WORD   EMS123,EMS132,EMS134,EMS123,0
073054  101452  101747  102016  EMT131: .WORD   EMS124,EMS132,EMS134,EMS123,0
073066  101504  101747  102016  EMT132: .WORD   EMS125,EMS132,EMS134,EMS123,0
073100  101547  101747  102016  EMT133: .WORD   EMS126,EMS132,EMS134,EMS123,0
073112  102147  101747  102016  EMT134: .WORD   EMS137,EMS132,EMS134,EMS123,0
073124  101651  101747  102016  EMT135: .WORD   EMS130,EMS132,EMS134,EMS123,0
073136  101707  101747  102016  EMT136: .WORD   EMS131,EMS132,EMS134,EMS123,0
073150  101307  101747  102060  EMT137: .WORD   EMS121,EMS132,EMS135,0
073160  101407  101747  102060  EMT140: .WORD   EMS123,EMS132,EMS135,0
073170  101307  101747  102122  EMT141: .WORD   EMS121,EMS132,EMS136,0
073200  102147  101747  102122  EMT142: .WORD   EMS137,EMS132,EMS136,0
073210  101452  101747  102122  EMT143: .WORD   EMS124,EMS132,EMS136,0
073220  101504  101747  102122  EMT144: .WORD   EMS125,EMS132,EMS136,0
073230  101547  101747  102122  EMT145: .WORD   EMS126,EMS132,EMS136,0
073240  101707  101747  102122  EMT146: .WORD   EMS131,EMS132,EMS136,0
073250  102656  101747  102122  EMT147: .WORD   EMS166,EMS132,EMS136,0
073260  101651  101747  102122  EMT150: .WORD   EMS130,EMS132,EMS136,0
073270  102204  101213  102225  EMT151: .WORD   EMS140,EMS115,EMS141,EMS70,0
073302  102253  101213  102270  EMT152: .WORD   EMS142,EMS115,EMS143,EMS72,0
073314  102316  102326  102355  EMT153: .WORD   EMS144,EMS145,EMS146,EMS115,EMS143,EMS72,0
073332  102316  102326  076634  EMT154: .WORD   EMS144,EMS145,EMS20,EMS115,EMS143,EMS70,0
073350  102411  102454  102522  EMT155: .WORD   EMS150,EMS152,EMS154,EMS153,0
073362  102374  102430  102522  EMT156: .WORD   EMS147,EMS151,EMS154,EMS155,0
```

```
073374  102374  102430  102556  EMT157: .WORD   EMS147,EMS151,EMS156,EMS157,0
073406  102626  102556  102611  EMT160: .WORD   EMS161,EMS156,EMS160,0
073416  076773  077031  102556  EMT161: .WORD   EMS25,EMS27,EMS156,EMS160,0
073430  077010  077031  102556  EMT162: .WORD   EMS26,EMS27,EMS156,EMS160,0
073442  077633  102711  102204  EMT163: .WORD   EMS47,EMS163,EMS140,0
073452  077633  076634  000000  EMT164: .WORD   EMS47,EMS20,0
073460  100034  076634  000000  EMT165: .WORD   EMS56,EMS20,0
073466  077604  076634  000000  EMT166: .WORD   EMS46,EMS20,0
073474  104205  076634  000000  EMT167: .WORD   EMS224,EMS20,0
073502  103057  102522  103032  EMT170: .WORD   EMS167,EMS154,EMS166,0
073512  103057  102522  102574  EMT171: .WORD   EMS167,EMS154,EMS157,0
073522  103057  102522  102536  EMT172: .WORD   EMS167,EMS154,EMS155,0
073532  103133  101747  101773  EMT173: .WORD   EMS171,EMS132,EMS133,0
073542  103133  101747  102016  EMT174: .WORD   EMS171,EMS132,EMS134,EMS123,0
073554  101137  103306  000000  EMT175: .WORD   EMS113,EMS177,0
073562  103330  103345  000000  EMT176: .WORD   EMS200,EMS201,0
073570  103443  102225  077042  EMT177: .WORD   EMS203,EMS141,EMS30,EMS202,0
073602  103443  077711  077042  EMT200: .WORD   EMS203,EMS51,EMS30,EMS202,0
073614  103443  103456  077042  EMT201: .WORD   EMS203,EMS204,EMS30,EMS202,0
073626  103443  077662  077042  EMT202: .WORD   EMS203,EMS50,EMS30,EMS202,0
073640  103443  102270  077042  EMT203: .WORD   EMS203,EMS143,EMS30,EMS202,0
073652  102411  100500  103413  EMT204: .WORD   EMS150,EMS74,EMS202,EMS115,EMS152,EMS72,0
073670  077662  100707  100453  EMT205: .WORD   EMS50,EMS103,EMS72,0
073700  100716  100463  103413  EMT206: .WORD   EMS104,EMS73,EMS202,0
073710  100517  102225  101213  EMT207: .WORD   EMS75,EMS141,EMS115,EMS140,0
073722  100517  102411  000000  EMT210: .WORD   EMS75,EMS150,0
073730  077711  100453  101213  EMT211: .WORD   EMS51,EMS72,EMS115,EMS50,EMS70,0
073744  102253  077751  101213  EMT212: .WORD   EMS142,EMS53,EMS115,EMS143,EMS72,0
073760  077662  100707  077751  EMT213: .WORD   EMS50,EMS103,EMS53,0
073770  077737  103504  103032  EMT214: .WORD   EMS52,EMS205,EMS166,EMS206,EMS115,EMS51,EMS72,0
074010  077737  101250  100034  EMT215: .WORD   EMS52,EMS117,EMS56,EMS163,0
074022  100034  077042  100245  EMT216: .WORD   EMS56,EMS30,EMS64,0
074032  077604  077042  100245  EMT217: .WORD   EMS46,EMS30,EMS64,0
074042  077052  077042  100245  EMT220: .WORD   EMS31,EMS30,EMS64,0
074052  077633  077042  100245  EMT221: .WORD   EMS47,EMS30,EMS64,0
074062  077737  101250  100556  EMT222: .WORD   EMS52,EMS117,EMS77,0
074072  077737  101250  100217  EMT223: .WORD   EMS52,EMS117,EMS63,0
074102  000000                  EMT224: .WORD
074104  000000                  EMT225: .WORD
074106  000000                  EMT226: .WORD
074110  000000                  EMT227: .WORD
074112  000000                  EMT230: .WORD
074114  000000                  EMT231: .WORD
074116  000000                  EMT232: .WORD
074120  000000                  EMT233: .WORD
074122  000000                  EMT234: .WORD
074124  000000                  EMT235: .WORD
074126  000000                  EMT236: .WORD
074130  000000                  EMT237: .WORD
074132  000000                  EMT240: .WORD
074134  000000                  EMT241: .WORD
074136  000000                  EMT242: .WORD
074140  000000                  EMT243: .WORD
074142  000000                  EMT244: .WORD
074144  000000                  EMT245: .WORD
074146  103057  101747  103543  EMT246: .WORD   EMS167,EMS132,EMS207,0
```

```
074156  103057  101747  103570  EMT247: .WORD   EMS167,EMS132,EMS210,EMS125,0
074170  103057  103601  103570  EMT250: .WORD   EMS167,EMS211,EMS210,EMS207,EMS206,0
074204  103617  103642  000000  EMT251: .WORD   EMS212,EMS213,0
074212  103016  103617  000000  EMT252: .WORD   EMS165,EMS212,0
074220  102374  102430  102556  EMT253: .WORD   EMS147,EMS151,EMS156,EMS210,EMS26,EMS27,0
074236  103443  100632  077042  EMT254: .WORD   EMS203,EMS101,EMS30,0
074246  103443  103057  077042  EMT255: .WORD   EMS203,EMS167,EMS30,0
074256  103443  100604  077042  EMT256: .WORD   EMS203,EMS100,EMS30,0
074266  076426  077604  077042  EMT257: .WORD   EMS11,EMS46,EMS30,EMS102,0
074300  077737  101250  100034  EMT260: .WORD   EMS52,EMS117,EMS56,EMS102,0
074312  077737  103504  104007  EMT261: .WORD   EMS52,EMS205,EMS220,EMS206,EMS115,EMS51,EMS72,0
074332  100716  100463  103413  EMT262: .WORD   EMS104,EMS73,EMS202,0
074342  077662  077751  100660  EMT263: .WORD   EMS50,EMS53,EMS102,0
074352  100034  077751  100660  EMT264: .WORD   EMS56,EMS53,EMS102,EMS115,EMS150,EMS152,EMS70,0
074372  100440  100034  100660  EMT265: .WORD   EMS71,EMS56,EMS102,EMS115,EMS150,EMS152,EMS72,0
074412  102253  102270  077751  EMT266: .WORD   EMS142,EMS143,EMS53,EMS102,0
074424  077662  102355  101213  EMT267: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,0
074442  077633  077751  100660  EMT270: .WORD   EMS47,EMS53,EMS102,EMS115,EMS140,0
074456  077633  077751  100660  EMT271: .WORD   EMS47,EMS53,EMS102,EMS115,EMS141,EMS72,0
074474  100517  102225  100660  EMT272: .WORD   EMS75,EMS141,EMS102,EMS115,EMS47,EMS73,0
074512  077711  100500  100660  EMT273: .WORD   EMS51,EMS74,EMS102,EMS115,EMS50,EMS70,0
074530  100217  077751  100111  EMT274: .WORD   EMS63,EMS53,EMS57,EMS115,EMS41,EMS146,0
074546  101220  077751  077362  EMT275: .WORD   EMS116,EMS53,EMS41,EMS57,0
074560  077633  077751  100111  EMT276: .WORD   EMS47,EMS53,EMS57,EMS115,EMS140,0
074574  077633  077751  100111  EMT277: .WORD   EMS47,EMS53,EMS57,EMS115,EMS141,EMS72,0
074612  100034  077751  100111  EMT300: .WORD   EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS70,0
074632  100440  100034  077751  EMT301: .WORD   EMS71,EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS72,0
074654  103016  100604  100707  EMT302: .WORD   EMS165,EMS100,EMS103,EMS57,0
074666  103016  100632  100707  EMT303: .WORD   EMS165,EMS101,EMS103,EMS57,0
074700  103016  100556  100707  EMT304: .WORD   EMS165,EMS77,EMS103,EMS57,0
074712  077604  077042  100245  EMT305: .WORD   EMS46,EMS30,EMS64,EMS57,0
074724  077662  077751  100111  EMT306: .WORD   EMS50,EMS53,EMS57,0
074734  077662  102355  101213  EMT307: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,EMS57,0
074754  102253  102270  077751  EMT310: .WORD   EMS142,EMS143,EMS53,EMS57,0
074766  100716  100707  077751  EMT311: .WORD   EMS104,EMS103,EMS53,EMS57,0
075000  100745  100707  077751  EMT312: .WORD   EMS105,EMS103,EMS53,EMS57,0
075012  102316  102326  100707  EMT313: .WORD   EMS144,EMS145,EMS103,EMS57,EMS115,EMS143,EMS70,0
075032  077410  100707  077751  EMT314: .WORD   EMS42,EMS103,EMS53,EMS57,0
075044  077052  100707  077751  EMT315: .WORD   EMS31,EMS103,EMS53,EMS57,0
075056  100440  077052  100707  EMT316: .WORD   EMS71,EMS31,EMS103,EMS57,0
075070  077437  100707  100111  EMT317: .WORD   EMS43,EMS103,EMS57,0
075100  077541  100707  100111  EMT320: .WORD   EMS45,EMS103,EMS57,0
075110  077466  100707  100111  EMT321: .WORD   EMS44,EMS103,EMS57,0
075120  100774  076634  000000  EMT322: .WORD   EMS106,EMS20,0
075126  077256  100707  100111  EMT323: .WORD   EMS36,EMS103,EMS57,0
075136  103206  077256  100707  EMT324: .WORD   EMS173,EMS36,EMS103,EMS57,0
075150  103166  077256  100707  EMT325: .WORD   EMS172,EMS36,EMS103,EMS57,0
075162  076500  103223  076522  EMT326: .WORD   EMS13,EMS174,EMS15,EMS35,EMS53,EMS175,0
075200  102374  102430  100500  EMT327: .WORD   EMS147,EMS151,EMS74,EMS175,0
075212  100331  077751  103231  EMT330: .WORD   EMS66,EMS53,EMS175,0
075222  077130  100707  077751  EMT331: .WORD   EMS33,EMS103,EMS53,EMS175,0
075234  077333  100707  077751  EMT332: .WORD   EMS40,EMS103,EMS53,EMS57,0
075246  077711  100500  100111  EMT333: .WORD   EMS51,EMS74,EMS57,EMS115,EMS50,EMS70,0
075264  100517  102225  100111  EMT334: .WORD   EMS75,EMS141,EMS57,EMS115,EMS47,EMS73,0
075302  100517  102411  102454  EMT335: .WORD   EMS75,EMS150,EMS152,EMS57,EMS115,EMS56,EMS73,0
075322  100137  100152  100201  EMT336: .WORD   EMS60,EMS61,EMS62,0
```

```
075332  101220  101250  077156  EMT337: .WORD   EMS116,EMS117,EMS34,0
075342  077156  077751  077763  EMT340: .WORD   EMS34,EMS53,EMS54,EMS111,0
075354  100005  077156  000000  EMT341: .WORD   EMS55,EMS34,0
075362  077737  101250  100034  EMT342: .WORD   EMS52,EMS117,EMS56,EMS57,0
075374  077737  101250  077541  EMT343: .WORD   EMS52,EMS117,EMS45,EMS57,0
075406  077737  101250  077466  EMT344: .WORD   EMS52,EMS117,EMS44,EMS57,0
075420  077737  101250  104027  EMT345: .WORD   EMS52,EMS117,EMS221,0
075430  100774  076634  101213  EMT346: .WORD   EMS106,EMS20,EMS115,EMS223,EMS72,0
075444  077737  103504  104077  EMT347: .WORD   EMS52,EMS205,EMS222,EMS206,0
075456  100517  102411  100660  EMT350: .WORD   EMS75,EMS150,EMS102,EMS115,EMS56,EMS73,0
075474  103057  100463  100660  EMT351: .WORD   EMS167,EMS73,EMS102,0
075504  103703  000000          EMT352: .WORD   EMS215,0
075510  103754  103443  103724  EMT353: .WORD   EMS217,EMS203,EMS216,0
075520  104027  076634  000000  EMT354: .WORD   EMS221,EMS20,0
```

```
075526  104257  105063  105140  EHT1:   .WORD   EH1,STSH1,STSH2,STSH4,0
075540  105063  105140  105265  EHT2:   .WORD   STSH1,STSH2,STSH4,0

075550  104276  000000          EHT110: .WORD   EH110,0
075554  104305  000000          EHT111: .WORD   EH111,0

075560  104324  000000          EHT114: .WORD   EH114,0
075564  104353  105063  105140  EHT223: .WORD   EH223,STSH1,STSH2,STSH4,0
075576  104401  105063  105140  EHT256: .WORD   EH256,STSH1,STSH2,STSH4,0

075610  104456  105063  105140  EHT336: .WORD   EH336,STSH1,STSH2,STSH4,0
075622  104515  105063  105140  EHT337: .WORD   EH337,STSH1,STSH2,STSH4,0
075634  104654  105063  105140  EHT344: .WORD   EH344,STSH1,STSH2,STSH4,0

075646  105014  000000          EHT353: .WORD   EH353,0
```

```
075652  105324  105420  105436  EDT1:    .WORD   ED1,STSD1,STSD2,STSD4
075662  105420  105436  105470  EDT2:    .WORD   STSD1,STSD2,STSD4

075670  105332                  EDT110:  .WORD   ED110
075672  105336                  EDT111:  .WORD   ED111

075674  105344                  EDT114:  .WORD   ED114
075676  105354  105420  105436  EDT223:  .WORD   ED223,STSD1,STSD2,STSD4

075706  105364  105420  105436  EDT336:  .WORD   ED336,STSD1,STSD2,STSD4
075716  105376  105420  105436  EDT337:  .WORD   ED337,STSD1,STSD2,STSD4
075726  105376  105420  105436  EDT344:  .WORD   ED337,STSD1,STSD2,STSD4,0

075740  105410                  EDT353:  .WORD   ED353
```

```
075742  105503  105521  105521  EFT1:   .WORD   EF111,STSF,STSF,STSF
075752  105521  105521  105521  EFT2:   .WORD   STSF,STSF,STSF

075760  105502                  EFT110: .WORD   EF110
075762  105503                  EFT111: .WORD   EF111

075764  105505                  EFT114: .WORD   EF114
075766  105505  105521  105521  EFT223: .WORD   EF114,STSF,STSF,STSF

075776  105510  105521  105521  EFT336: .WORD   EF336,STSF,STSF,STSF
076006  105510  105521  105521  EFT337: .WORD   EF336,STSF,STSF,STSF
076016  105510  105521  105521  EFT344: .WORD   EF336,STSF,STSF,STSF

076026  105505                  EFT353: .WORD   EF114
```

```
                        .SBTTL  ERROR MESSAGE STRINGS

076030 051127 047117 020107 EMS1:   .ASCIZ  @WRONG UNIT SELECTED (RMCS2, BITS 0-2) @
076077    104 053105 041511 EMS2:   .ASCIZ  @DEVICE WENT @
076114 047125 053101 044501 EMS3:   .ASCIZ  @UNAVAILABLE "DVA" (RMCS1, BIT 11) @
076157    116 047117 054105 EMS4:   .ASCIZ  @NONEXISTENT "NED" (RMCS2, BIT 12) @
076222 047503 046515 047101 EMS5:   .ASCIZ  @COMMAND NOT COMPLETED @
076252 047503 052116 047522 EMS6:   .ASCIZ  @CONTROLLER NOT READY (RMCS1, BIT 7) @
076317    104 044522 042526 EMS7:   .ASCIZ  @DRIVE NOT READY "DRY" (RMDS, BIT 7) @
076364 047507 047040 052117 EMS10:  .ASCIZ  @GO NOT RESET "GO" (RMCS1, BIT 0) @
076426 047111 040526 044514 EMS11:  .ASCIZ  @INVALID @
076437    106 047125 052103 EMS12:  .ASCIZ  @FUNCTION CODE (RMCS1, BITS 1-5) @
076500 040515 051523 052502 EMS13:  .ASCIZ  @MASSBUS @
076511    103 047117 051124 EMS14:  .ASCIZ  @CONTROL @
076522 052502 020123 040520 EMS15:  .ASCIZ  @BUS PARITY ERROR @
076544 046442 050103 021105 EMS16:  .ASCIZ  @"MCPE" (RMCS1, BIT 13) @
076574 051124 047101 043123 EMS17:  .ASCIZ  @TRANSFER ERROR (RMCS1, BIT 14) @
076634 044123 052517 042114 EMS20:  .ASCIZ  @SHOULD NOT BE SET @
076657    127 051117 020104 EMS21:  .ASCIZ  @WORD COUNT (RMWC) @
076702 052502 020123 051050 EMS22:  .ASCIZ  @BUS (RMBA) @
076716 046042 052102 020042 EMS23:  .ASCIZ  @"LBT" (RMDS, BIT 10) @
076744 040442 042517 020042 EMS24:  .ASCIZ  @"AOE" (RMER1, BIT 09) @
076773    104 051511 020113 EMS25:  .ASCIZ  @DISK (RMDA) @
077010 054503 044514 042116 EMS26:  .ASCIZ  @CYLINDER (RMDC) @
077031    101 042104 042522 EMS27:  .ASCIZ  @ADDRESS @
077042 052123 052101 051525 EMS30:  .ASCIZ  @STATUS @
077052 053442 042514 020042 EMS31:  .ASCIZ  @"WLE" (RMER1, BIT 11) @
077101    042 050125 021105 EMS32:  .ASCIZ  @"UPE" (RMCS2, BIT 13) @
077130 053442 043103 020042 EMS33:  .ASCIZ  @"WCF" (RMER1, BIT 5) @
077156 051127 052111 020105 EMS34:  .ASCIZ  @WRITE CHECK ERROR-"WCE" (RMCS2, BIT 14) @
077227    042 042115 042520 EMS35:  .ASCIZ  @"MDPE" (RMCS2, BIT 8) @
077256 042042 045503 020042 EMS36:  .ASCIZ  @"DCK" (RMER1, BIT 15) @
077305    042 041505 021110 EMS37:  .ASCIZ  @"ECH" (RMER1, BIT 6) @
077333    042 046104 021124 EMS40:  .ASCIZ  @"DLT" (RMCS2, BIT 15) @
077362 046442 043130 020042 EMS41:  .ASCIZ  @"MXF" (RMCS2, BIT 9) @
077410 042042 042524 020042 EMS42:  .ASCIZ  @"DTE" (RMER1, BIT 12) @
077437    042 041510 041522 EMS43:  .ASCIZ  @"HCRC" (RMER1, BIT 8) @
077466 042510 042101 051105 EMS44:  .ASCIZ  @HEADER COMPARE ERROR "HCE" (RMER1, BIT 7) @
077541    106 051117 040515 EMS45:  .ASCIZ  @FORMAT ERROR "FER" (RMER1, BIT 4) @
077604 044442 042501 020042 EMS46:  .ASCIZ  @"IAE" (RMER1, BIT 10) @
077633    042 050117 021111 EMS47:  .ASCIZ  @"OPI" (RMER1, BIT 13) @
077662 051442 044513 020042 EMS50:  .ASCIZ  @"SKI" (RMER2, BIT 14) @
077711    042 044520 021120 EMS51:  .ASCIZ  @"PIP" (RMDS, BIT 13) @
077737    124 042510 051040 EMS52:  .ASCIZ  @THE RM03 @
077751    104 052105 041505 EMS53:  .ASCIZ  @DETECTED @
077763    101 020124 047101 EMS54:  .ASCIZ  @AT AN UNEXPECTED @
100005    111 041516 051117 EMS55:  .ASCIZ  @INCORRECT DATA DURING @
100034 047111 040526 044514 EMS56:  .ASCIZ  @INVALID COMMAND ERROR "IVC" (RMER2, BIT 12) @
100111    104 051125 047111 EMS57:  .ASCIZ  @DURING DATA TRANSFER @
100137    104 052101 020101 EMS60:  .ASCIZ  @DATA READ @
100152 047504 051505 047040 EMS61:  .ASCIZ  @DOES NOT COMPARE WITH @
100201    104 052101 020101 EMS62:  .ASCIZ  @DATA WRITTEN @
100217    042 040520 021122 EMS63:  .ASCIZ  @"PAR" (RMER1, BIT 3) @
100245    111 020123 047111 EMS64:  .ASCIZ  @IS INCORRECT @
100263    103 046517 047520 EMS65:  .ASCIZ  @COMPOSITE ERROR "ERR" (RMDS, BIT 14) @
100331    104 052101 020101 EMS66:  .ASCIZ  @DATA PARITY ERROR "DPE" (RMER2, BIT 3) @
```

```
100401     104  051125  047111  EMS67:  .ASCIZ  @DURING SEEK COMMAND @
100426  051511  051040  051505  EMS70:  .ASCIZ  @IS RESET @
100440  051105  047522  042516  EMS71:  .ASCIZ  @ERRONEOUS @
100453     111  020123  042523  EMS72:  .ASCIZ  @IS SET @
100463     104  042111  047040  EMS73:  .ASCIZ  @DID NOT SET @
100500  044504  020104  047516  EMS74:  .ASCIZ  @DID NOT RESET @
100517     114  051517  020124  EMS75:  .ASCIZ  @LOST @
100525     104  051125  047111  EMS76:  .ASCIZ  @DURING PACK ACK COMMAND @
100556  051042  051115  020042  EMS77:  .ASCIZ  @"RMR" (RMER1, BIT 2) @
100604  044442  051114  020042  EMS100: .ASCIZ  @"ILR" (RMER1, BIT 1) @
100632  044442  043114  020042  EMS101: .ASCIZ  @"ILF" (RMER1, BIT 0) @
100660  052504  044522  043516  EMS102: .ASCIZ  @DURING SEARCH COMMAND @
100707     105  051122  051117  EMS103: .ASCIZ  @ERROR @
100716  046042  041502  020042  EMS104: .ASCIZ  @"LBC" (RMER2, BIT 10) @
100745     042  051514  021103  EMS105: .ASCIZ  @"LSC" (RMER2, BIT 11) @
100774  042510  042101  051105  EMS106: .ASCIZ  @HEADER ERRORS @
101013     102  051525  052040  EMS110: .ASCIZ  @BUS TIMEOUT (04 TRAP) @
101042  042101  051104  051505  EMS111: .ASCIZ  @ADDRESS @
101053     127  042510  020116  EMS112: .ASCII  @WHEN READING/WRITING RH REGISTERS @
101115     101  020124  044124          .ASCIZ  @AT THE FOLLOWING @
101137     124  042510  051440  EMS113: .ASCIZ  @THE SELECTED DEVICE IS @
101167     116  047117  054105  EMS114: .ASCIZ  @NONEXISTENT DEVICE @
101213     040  006455  000012  EMS115: .ASCIZ  @ -@<CR><LF>
101220  044124  020105  040515  EMS116: .ASCIZ  @THE MASSBUS CONTROLLER @
101250  040506  046111  042105  EMS117: .ASCIZ  @FAILED TO DETECT @
101272  047516  020124  047101  EMS120: .ASCIZ  @NOT AN RM03 @
101307     103  047117  051124  EMS121: .ASCIZ  @CONTROL STATUS REGISTER 1, RMCS1, @
101352  052502  020123  042101  EMS122: .ASCIZ  @BUS ADDRESS REGISTER, RMBA, @
101407     103  047117  051124  EMS123: .ASCIZ  @CONTROL STATUS REGISTER 2, RMCS2, @
101452  051105  047522  020122  EMS124: .ASCIZ  @ERROR REGISTER 1, RMER1, @
101504  052101  042524  052116  EMS125: .ASCIZ  @ATTENTION SUMMARY REGISTER, RMAS, @
101547     115  044501  052116  EMS126: .ASCIZ  @MAINTENANCE REGISTER #1, RMMR #1, @
101612  041505  020103  047520  EMS127: .ASCIZ  @ECC POSITION REGISTER, RMEC1, @
101651     105  041503  050040  EMS130: .ASCIZ  @ECC PATTERN REGISTER, RMEC2, @
101707     115  044501  052116  EMS131: .ASCIZ  @MAINTENANCE REGISTER 2, RMMR2, @
101747     116  052117  044440  EMS132: .ASCIZ  @NOT INITIALIZED BY @
101773     125  044516  052502  EMS133: .ASCIZ  @UNIBUS INITIALIZE @
102016  047503  052116  047522  EMS134: .ASCIZ  @CONTROLLER CLEAR, I.E. BIT 5 OF @
102060  044122  030461  042440  EMS135: .ASCIZ  @RH11 ERROR CLEAR (RMCS1, BIT 14) @
102122  051104  053111  020105  EMS136: .ASCIZ  @DRIVE CLEAR COMMAND @
102147     104  044522  042526  EMS137: .ASCIZ  @DRIVE STATUS REGISTER, RMDS @
102204  042515  044504  046525  EMS140: .ASCIZ  @MEDIUM OFF LINE @
102225     042  047515  021114  EMS141: .ASCIZ  @"MOL" (RMDS, BIT 12) @
102253     104  044522  042526  EMS142: .ASCIZ  @DRIVE FAULT @
102270  042042  041526  020042  EMS143: .ASCIZ  @"DVC" (RMER2, BIT 7) @
102316  047125  040523  042506  EMS144: .ASCIZ  @UNSAFE @
102326  052442  051516  020042  EMS145: .ASCIZ  @"UNS" (RMER1, BIT 14) @
102355     123  047510  046125  EMS146: .ASCIZ  @SHOULD BE SET @
102374  043117  051506  052105  EMS147: .ASCIZ  @OFFSET MODE @
102411     040  047526  052514  EMS150: .ASCIZ  @ VOLUME VALID @
102430  047442  021115  024040  EMS151: .ASCIZ  @"OM" (RMDS, BIT 0) @
102454  053042  021126  024040  EMS152: .ASCIZ  @"VV" (RMDS, BIT 6) @
102500  040520  045503  040440  EMS153: .ASCIZ  @PACK ACK COMMAND @
102522  047516  020124  042523  EMS154: .ASCIZ  @NOT SET BY @
102536  043117  051506  052105  EMS155: .ASCIZ  @OFFSET COMMAND @
102556  047516  020124  042522  EMS156: .ASCIZ  @NOT RESET BY @
```

```
102574  052122  020103  047503  EMS157: .ASCIZ  @RTC COMMAND @
102611     122  050111  041440  EMS160: .ASCIZ  @RIP COMMAND @
102626  043117  051506  052105  EMS161: .ASCIZ  @OFFSET REGISTER (RMOF) @
102656  051105  047522  020122  EMS162: .ASCIZ  @ERROR REGISTER #2, RMER2, @
102711     104  051125  047111  EMS163: .ASCIZ  @DURING RECALIBRATE @
102735     111  020123  047111  EMS164: .ASCII  @IS INTERMITTENT OR DRIVE DIDNT DROP ON @
103004  054503  044514  042116          .ASCIZ  @CYLINDER @
103016  047125  054105  042520  EMS165: .ASCIZ  @UNEXPECTED @
103032  042522  040503  044514  EMS166: .ASCIZ  @RECALIBRATE COMMAND @
103057     042  052101  021101  EMS167: .ASCIZ  @"ATA" (RMDS, BIT15) @
103104  044127  047105  051040  EMS170: .ASCIZ  @WHEN READING REGISTER @
103133     105  051122  051117  EMS171: .ASCIZ  @ERROR REGISTER #2, RMER2, @
103166  047516  051116  041505  EMS172: .ASCIZ  @NONRECOVERABLE @
103206  042522  047503  042526  EMS173: .ASCIZ  @RECOVERABLE @
103223     104  052101  020101  EMS174: .ASCIZ  @DATA @
103231     104  051125  047111  EMS175: .ASCIZ  @DURING WRITE COMMAND @
103257     042  050117  021105  EMS176: .ASCIZ  @"OPE" (RMER2, BIT 13) @
103306  047111  053440  044522  EMS177: .ASCIZ  @IN WRITE PROTECT @
103330  040503  020116  047516  EMS200: .ASCIZ  @CAN NOT SET @
103345     104  040511  047107  EMS201: .ASCIZ  @DIAGNOSTIC MODE "DMD" (RMMR1, BIT 0) @
103413     104  051125  047111  EMS202: .ASCIZ  @DURING DIAGNOSTIC MODE @
103443     111  041516  051117  EMS203: .ASCIZ  @INCORRECT @
103456  053442  046122  020042  EMS204: .ASCIZ  @"WRL" (RMDS, BIT 11) @
103504  054105  041505  052125  EMS205: .ASCIZ  @EXECUTED @
103516  044527  044124  041440  EMS206: .ASCIZ  @WITH COMP ERROR SET @
103543     042  047507  020042  EMS207: .ASCIZ  @"GO" (RMCS1, BIT 0) @
103570  051127  052111  047111  EMS210: .ASCIZ  @WRITING @
103601     127  051501  051040  EMS211: .ASCIZ  @WAS RESET BY @
103617     120  047522  051107  EMS212: .ASCIZ  @PROGRAM INTERRUPT @
103642  040527  020123  047516  EMS213: .ASCIZ  @WAS NOT GENERATED @
103665     123  042505  020113  EMS214: .ASCIZ  @SEEK COMMAND @
103703     120  047522  051107  EMS215: .ASCIZ  @PROGRAM TIMEOUT @
103724  052504  044522  043516  EMS216: .ASCIZ  @DURING LOOK AHEAD TEST @
103754  047514  045517  040440  EMS217: .ASCIZ  @LOOK AHEAD REGISTER,RMLA, @
104007     123  040505  041522  EMS220: .ASCIZ  @SEARCH COMMAND @
104027     102  042101  051440  EMS221: .ASCIZ  @BAD SECTOR ERROR "BSE" (RMER2, BIT 15) @
104077     101  042040  052101  EMS222: .ASCIZ  @A DATA TRANSFER COMMAND @
104130  042510  042101  051105  EMS223: .ASCIZ  @HEADER COMPARE INHIBIT "HCI" (RMOF, BIT 10) @
104205     116  047117  054105  EMS224: .ASCIZ  @NONEXISTENT MEMORY "NEM" (RMCS2, BIT 11) @
```

```
104257    105  050130  052103  EH1:    .ASCIZ  @EXPCTD  RECEVD@
104276 052502  040523  051104  EH110:  .ASCIZ  @BUSADR@
104305    040  046522  051503  EH111:  .ASCIZ  @ RMCS2   RMCS1@

104324 042522  042503  042126  EH114:  .ASCIZ  @RECEVD  SNGPRT  DULPRT@
104353    105  050130  052103  EH223:  .ASCIZ  @EXPCTD  RECEVD   DATA@
104401    105  050130  052103  EH256:  .ASCII  @EXPCTD  RECEVD  RGSTR@<CR><LF>
104430 052123  052101  051525          .ASCIZ  @STATUS  STATUS  INDEX@

104456 042107  042101  051522  EH336:  .ASCIZ  @GDADRS  GDDATA  BDADRS  BDDATA@
104515    122  041515  031123  EH337:  .ASCII  @RMCS2   STATUS  FAILING   DATA@<CR><LF>
104555    137  057537  057537          .ASCII  @++++++++++++++++  ++++++++++++++++@<CR><LF>
104615    105  050130  052103          .ASCIZ  @EXPCTD  RECEVD    BIT    ADRESS@

104654 046522  051105  020061  EH344:  .ASCII  @RMER1   STATUS  HEADER  FAILING@<CR><LF>
104715    137  057537  057537          .ASCII  @++++++++++++++++   WORD     BIT@<CR><LF>
104754 054105  041520  042124          .ASCIZ  @EXPCTD  RECEVD  NUMBER  POSITON@
105014 054105  041520  042124  EH353:  .ASCII  @EXPCTD  RECEVD@<CR><LF>
105034 051040  046115  020101          .ASCIZ  @ RMLA    RMLA    RMOF @

105063    040  046522  051503  STSH1:  .ASCII  @ RMCS1  RMCS2   RMDS    RMER1   RMER2@
105130 020040  051040  040515          .ASCIZ  @   RMAS@
105140 051040  053515  020103  STSH2:  .ASCII  @ RMWC    RMBA    RMDA    RMOF    RMDC@
105205    040  020040  051040          .ASCIZ  @    RMEC1   RMEC2@
105227    040  046522  040504  STSH3:  .ASCIZ  @ RMDA    RMDC    RMOF    RMLA@
105265    040  046522  051115  STSH4:  .ASCIZ  @ RMMR1   RMMR2   RMDT    RMSN@
```

```
                105324                          .EVEN
105324  001140  001142  000000  ED1:    .WORD   $GDDAT,$BDDAT,0
105332  001276  000000          ED110:  .WORD   $BASE,0
105336  001174  001176  000000  ED111:  .WORD   $TMP0,$TMP1,0

105344  001356  001176  001200  ED114:  .WORD   RMDTI,$TMP1,$TMP2,0
105354  001140  001142  001174  ED223:  .WORD   $GDDAT,$BDDAT,$TMP0,0

105364  001134  001140  001136  ED336:  .WORD   $GDADR,$GDDAT,$BDADR,$BDDAT,0

105376  001140  001142  001174  ED337:  .WORD   $GDDAT,$BDDAT,$TMP0,$TMP1,0
105410  001140  001142  001432  ED353:  .WORD   $GDDAT,$BDDAT,RMOF0,0

105420  001330  001340  001342  STSD1:  .WORD   RMCS1I,RMCS2I,RMDSI,RMER1I,RMER2I,RMASI,0
105436  001332  001334  001336  STSD2:  .WORD   RMWCI,RMBAI,RMDAI,RMOFI,RMDCI,RMEC1I
105452  001376  000000          .WORD   RMEC2I,0
105456  001336  001364  001362  STSD3:  .WORD   RMDAI,RMDCI,RMOFI,RMLAI,0
105470  001354  001370  001356  STSD4:  .WORD   RMMR1I,RMMR2I,RMDTI,RMSNI,0
```

```
105502  000             EF110:  .BYTE   0
105503  000     000     EF111:  .BYTE   0,0
105505  000     000     000     EF114:  .BYTE   0,0,0
105510  000     000     000     EF336:  .BYTE   0,0,0,0
105514  000     000     000     EF337:  .BYTE   0,0,0,0,0

105521  000     000     000     STSF:   .BYTE   0,0,0,0,0,0,0
```

```
105530  000402                  .EVEN
106534  177777          MFGFIL: .BLKW   258.
106536  177777                  .WORD   -1
106540  000402                  .WORD   -1
107544  177777          USRFIL: .BLKW   258.
107546  177777                  .WORD   -1
                                .WORD   -1
                        .EVEN
```

```
                        107550                          BUFFER:
107550  000402                                  BUFONE: .BLKW   258.
110554  000402                                  BUFTWO: .BLKW   258.


                        107550                  .       =       BUFFER
107550                                  HELP:
107550  005015                          .ASCII  <CR><LF>
                                        ;.ASCII @CAUTION: PARTS 2 AND 3 OF THE FUNCTIONAL TEST LEAVE BAD@<CR><LF>
                                        ;.ASCII @HEADERS ON THE PACK SO BE SURE TO FORMAT THE PACK WHEN@<CR><LF>
                                        ;.ASCII @DONE.@<CR><LF>
107552  046011  051511  020124         .ASCII  @       LIST OF TESTS@<CR><LF>
107572  057411  057537  057537         .ASCII  @       ++++++++++++++@<CR><LF>
107612  030524  041411  047117         .ASCII  @T1     CONTROLLER ACCESS TEST@<CR><LF>
107645     124  004462  042504         .ASCII  @T2     DEVICE AVAILABLE TEST@<CR><LF>
107677     124  004463  051104         .ASCII  @T3     DRIVE TYPE TEST@<CR><LF>
107723     124  004464  051127         .ASCII  @T4     WRITE, READ ZEROS@<CR><LF>
107751     124  004465  051127         .ASCII  @T5     WRITE, WRITE CHECK ZEROS@<CR><LF>
110006  033124  053411  044522         .ASCII  @T6     WRITE, WRITE CHECK ZEROS W/ WCE ERROR@<CR><LF>
110060  033524  053411  044522         .ASCII  @T7     WRITE, READ ONES@<CR><LF>
110105     124  030061  053411         .ASCII  @T10    WRITE, WRITE CHECK ONES@<CR><LF>
110142  030524  004461  051127         .ASCII  @T11    WRITE, WRITE CHECK ONES W/ WCE ERROR@<CR><LF>
110214  030524  004462  051127         .ASCII  @T12    WRITE, WRITE CHECK MULTIPLE SECTORS@<CR><LF>
110265     124  031461  053411         .ASCII  @T13    WRITE, READ W/ IMPLIED SEEK@<CR><LF>
110326  030524  004464  051127         .ASCII  @T14    WRITE, WRITE CHECK W/ HEAD SWITCHING@<CR><LF>
110400  030524  004465  051127         .ASCII  @T15    WRITE, WRITE CHECK W/ MID-TRANSFER SEEK@<CR><LF>
110455     124  033061  053411         .ASCII  @T16    WRITE, READ W/ HCE ERROR@<CR><LF>
110513     124  033461  053411         .ASCII  @T17    WRITE, READ W/ HCI@<CR><LF>
110543     124  030062  053411         .ASCII  @T20    WRITE, READ W/ IVC ERROR@<CR><LF>
110601     124  030462  053411         .ASCII  @T21    WRITE, READ W/ ABORT@<CR><LF>
110633     124  031062  053411         .ASCII  @T22    WRITE, READ W/ BAI@<CR><LF>
110663     124  031462  053411         .ASCII  @T23    WRITE, READ EACH CURRENT LEVEL@<CR><LF>
110727     124  032062  053411         .ASCII  @T24    WRITE, WRITE CHECK W/ CURRENT LEVEL SWITCHING@<CR><LF>
111012  031124  004465  051127         .ASCII  @T25    WRITE, READ EARLY PEAK SHIFT@<CR><LF>
111054  031124  004466  051127         .ASCII  @T26    WRITE, READ MIXED PEAK SHIFT@<CR><LF>
111116  031124  004467  051127         .ASCII  @T27    WRITE, READ EACH TRACK@<CR><LF>
111152  005015                          .ASCII  <CR><LF>
111154  047411  042520  040522         .ASCII  @       OPERATIONAL SWITCH SETTINGS@<CR><LF>
111212  057411  057537  057537         .ASCII  @       ++++++++++++++++++++++++++@<CR><LF>
111250  005015                          .ASCII  <CR><LF>
111252  053523  052111  044103         .ASCII  @SWITCH                 USE@<CR><LF>
111270  026455  026455  026455         .ASCII  @------                 ---------------@<CR><LF>
111326  020040  032461  004411         .ASCII  @   15                  HALT ON ERROR@<CR><LF>
111353     040  030440  004464         .ASCII  @   14                  LOOP ON TEST@<CR><LF>
111377     040  030440  004463         .ASCII  @   13                  INHIBIT ERROR TYPEOUTS@<CR><LF>
111435     040  030440  004462         .ASCII  @   12                  @<CR><LF>
111445     040  030440  004461         .ASCII  @   11                  INHIBIT ITERATIONS@<CR><LF>
111477     040  030440  004460         .ASCII  @   10                  BELL ON ERROR@<CR><LF>
111524  020040  034440  004411         .ASCII  @    9                  LOOP ON ERROR@<CR><LF>
111551     040  020040  004470         .ASCII  @    8                  LOOP ON TEST IN SWR<7:0>@<CR><LF>
111611     040  020040  004467         .ASCII  @    7                  TN128@<CR><LF>
111626  020040  033040  004411         .ASCII  @    6                  TN64@<CR><LF>
111642  020040  032440  004411         .ASCII  @    5                  TN32@<CR><LF>
111656  020040  032040  004411         .ASCII  @    4                  TN16@<CR><LF>
111672  020040  031440  004411         .ASCII  @    3                  TN8@<CR><LF>
111705     040  020040  004462         .ASCII  @    2                  TN4@<CR><LF>
```

```
111720  020040  030440  004411  .ASCII  @   1          TN2@<CR><LF>
111733     040  020040  004460  .ASCII  @   0          TN1@<CR><LF>
111746  005015                  .ASCII  <CR><LF>
111750  005015     000          .ASCIZ  <CR><LF>
        000001                  .END
```

```
ABASE = 176700          1208#   1316    1357
ACDW1 = 000000          1316    1359
ACDW2 = 000000          1316    1360
ACKSTS  053676          9369    9557   11612#
ACPUOP= 000000          1316    1331
ADDW0 = 000000          1316    1361
ADDW1 = 000000          1316    1362
ADDW10= 000000          1316
ADDW11= 000000          1316
ADDW12= 000000          1316
ADDW13= 000000          1316
ADDW14= 000000          1316
ADDW15= 000000          1316
ADDW2 = 000000          1316    1363
ADDW3 = 000000          1316    1364
ADDW4 = 000000          1316    1365
ADDW5 = 000000          1316    1366
ADDW6 = 000000          1316    1367
ADDW7 = 000000          1316    1368
ADDW8 = 000000          1316
ADDW9 = 000000          1316
ADEVCT= 000000          1316    1322
ADEVM = 000000          1316    1358
ADR   = 000001          3480#   3481    3522#   3523#   3524    3566#   3567#   3568    3604#   3605#   3606    3861#   3862#
                        3863    4109#   4110#   4111    4401#   4402#   4403    4658#   4659#   4660    4906#   4907#   4908
                        5198#   5199#   5200    5451#   5452#   5453    5767#   5768#   5769    6020#   6021#   6022    6273#
                        6274#   6275    6684#   6685#   6686    7019#   7020#   7021    7257#   7258#   7259    7469#   7470#
                        7471    7734#   7735#   7736    7998#   7999#   8000    8246#   8247#   8248    8503#   8504#   8505
                        8760#   8761#   8762
AENV  = 000000          1316    1327
AENVM = 000000          1316    1328
AFATAL= 000000          1316    1319
AMADR1= 000000          1316    1344
AMADR2= 000000          1316    1348
AMADR3= 000000          1316    1351
AMADR4= 000000          1316    1354
AMAMS1= 000000          1316    1338
AMAMS2= 000000          1316    1346
AMAMS3= 000000          1316    1349
AMAMS4= 000000          1316    1352
AMSGAD= 000000          1316    1324
AMSGLG= 000000          1316    1325
AMSGTY= 000000          1316    1318
AMTYP1= 000000          1316    1339
AMTYP2= 000000          1316    1347
AMTYP3= 000000          1316    1350
AMTYP4= 000000          1316    1353
AOE   = 001000           988#    999    10782   10797   10812   10827   10831   11078   11082   13835
APASS = 000000          1316    1321
APE   = 100000          1187#
APRIOR= 000000          1316
APTCSU= 000040         13188   13833#
APTENV= 000001         13181   13367   13789   13831#
APTSIZ= 000200          3224   13830#
APTSPO= 000100         13183   13791   13832#
ARGS  = 000006          3617#   3635#   3657#   3665#   3668#   3671#   3679#   3699#   3723#   3731#   3734#   3737#   3745#
```

```
                3752#  3761#  3780#  3802#  3810#  3813#  3816#  3824#  3831#  3840#  3847#  3874#  3892#
3914#  3922#  3925#  3928#  3936#  3956#  3980#  3988#  3991#  3994#  4002#  4009#  4018#
4037#  4058#  4066#  4069#  4072#  4080#  4087#  4096#  4122#  4140#  4162#  4170#  4173#
4176#  4184#  4204#  4228#  4236#  4239#  4242#  4250#  4257#  4266#  4288#  4309#  4317#
4320#  4323#  4331#  4340#  4381#  4414#  4432#  4454#  4462#  4465#  4468#  4476#  4496#
4520#  4528#  4531#  4534#  4542#  4549#  4558#  4577#  4599#  4607#  4610#  4613#  4621#
4628#  4637#  4644#  4671#  4689#  4711#  4719#  4722#  4725#  4733#  4753#  4777#  4785#
4788#  4791#  4799#  4806#  4815#  4834#  4855#  4863#  4866#  4869#  4877#  4884#  4893#
4919#  4937#  4959#  4967#  4970#  4973#  4981#  5001#  5025#  5033#  5036#  5039#  5047#
5054#  5063#  5085#  5106#  5114#  5117#  5120#  5128#  5137#  5178#  5211#  5229#  5251#
5259#  5262#  5265#  5273#  5300#  5322#  5330#  5333#  5336#  5344#  5351#  5360#  5379#
5400#  5408#  5411#  5414#  5422#  5429#  5438#  5464#  5482#  5504#  5512#  5515#  5518#
5526#  5549#  5560#  5568#  5569#  5570#  5576#  5600#  5608#  5611#  5619#  5626#  5635#
5657#  5668#  5676#  5677#  5678#  5684#  5707#  5716#  5719#  5727#  5734#  5743#  5750#
5780#  5798#  5820#  5828#  5831#  5834#  5842#  5869#  5891#  5899#  5902#  5905#  5913#
5920#  5929#  5948#  5969#  5977#  5980#  5983#  5991#  5998#  6007#  6033#  6051#  6073#
6081#  6084#  6087#  6095#  6122#  6144#  6152#  6155#  6158#  6166#  6173#  6182#  6201#
6222#  6230#  6233#  6236#  6244#  6251#  6260#  6290#  6308#  6337#  6345#  6348#  6351#
6359#  6379#  6403#  6411#  6414#  6417#  6425#  6446#  6465#  6482#  6494#  6512#  6535#
6543#  6546#  6549#  6557#  6577#  6596#  6613#  6625#  6654#  6699#  6717#  6746#  6754#
6757#  6760#  6768#  6788#  6812#  6820#  6823#  6826#  6834#  6854#  6863#  6881#  6903#
6911#  6914#  6917#  6925#  6945#  6954#  6961#  6990#  7035#  7050#  7092#  7100#  7103#
7106#  7114#  7123#  7137#  7156#  7197#  7205#  7208#  7211#  7219#  7228#  7242#  7274#
7288#  7322#  7323#  7329#  7343#  7346#  7354#  7362#  7381#  7413#  7415#  7423#  7438#
7441#  7449#  7456#  7482#  7500#  7522#  7530#  7533#  7536#  7544#  7564#  7593#  7601#
7604#  7607#  7615#  7622#  7631#  7650#  7675#  7683#  7686#  7689#  7697#  7704#  7713#
7720#  7747#  7765#  7787#  7795#  7798#  7801#  7809#  7829#  7853#  7861#  7864#  7867#
7875#  7882#  7891#  7910#  7933#  7941#  7944#  7947#  7955#  7962#  7971#  7978#  8011#
8029#  8051#  8059#  8062#  8065#  8073#  8093#  8117#  8125#  8128#  8131#  8139#  8146#
8155#  8174#  8195#  8203#  8206#  8209#  8217#  8224#  8233#  8259#  8277#  8299#  83C7#
8310#  8313#  8321#  8341#  8365#  8373#  8376#  8379#  8387#  8394#  8403#  8422#  8444#
8452#  8455#  8458#  8466#  8473#  8482#  8489#  8516#  8534#  8556#  8564#  8567#  8570#
8578#  8598#  8622#  8630#  8633#  8636#  8644#  8651#  8660#  8679#  8701#  8709#  8712#
8715#  8723#  8730#  8739#  8746#  8773#  8791#  8813#  8821#  8824#  8827#  8835#  8855#
8879#  8887#  8890#  8893#  8901#  8908#  8917#  8936#  8958#  8966#  8969#  8972#  8980#
8987#  8996#  9003#

ASNDA   001500  1440#  6647   6984   9747*  9776   9822*  9823   9825*  9826*  9827   9829*  9874
ASNDC   001476  1439#  6646   6983   9746*  9775   9830*  9831   9833*  9873
ASWREG= 000000  1316   1329
ATA     = 100000   968#  10609  10610  10612  11344  11345  11380  11383  11875  11876  11915  11918  13835
ATESTN= 000000  1316   1320
ATNMSK= 000377  1005#  12072
ATNTBL  070776  3422   3440   13835#
AUNIT   = 000000  1316   1323
AUSWR   = 000000  1316   1330
AVECT1= 120254  1209#  1316   1355
AVECT2= 000000  1316   1356
A16     = 000400  1158#
A17     = 001000  1157#
BACK    = 000000  3617#  3622   3635#  3639   3657#  3661   3665#  3668#  3671#  3675   3679#  3683   3699#
3704   3723#  3727   3731#  3734#  3737#  3741   3745#  3749   3752#  3756   3761#  3765
3780#  3785   3802#  3806   3810#  3813#  3816#  3820   3824#  3828   3831#  3835   3840#
3844   3847#  3853   3874#  3879   3892#  3896   3914#  3918   3922#  3925#  3928#  3932
3936#  3940   3956#  3961   3980#  3984   3988#  3991#  3994#  3998   4002#  4006   4009#
4013   4018#  4022   4037#  4042   4058#  4062   4066#  4069#  4072#  4076   4080#  4084
4087#  4091   4096#  4100   4122#  4127   4140#  4144   4162#  4166   4170#  4173#  4176#
```

```
4180    4184#   4188    4204#   4209    4228#   4232    4236#   4239#   4242#   4246    4250#   4254
4257#   4261    4266#   4270    4288#   4293    4309#   4313    4317#   4320#   4323#   4327    4331#
4335    4340#   4344    4381#   4385    4414#   4419    4432#   4436    4454#   4458    4462#   4465#
4468#   4472    4476#   4480    4496#   4501    4520#   4524    4528#   4531#   4534#   4538    4542#
4546    4549#   4553    4558#   4562    4577#   4582    4599#   4603    4607#   4610#   4613#   4617
4621#   4625    4628#   4632    4637#   4641    4644#   4650    4671#   4676    4689#   4693    4711#
4715    4719#   4722#   4725#   4729    4733#   4737    4753#   4758    4777#   4781    4785#   4788#
4791#   4795    4799#   4803    4806#   4810    4815#   4819    4834#   4839    4855#   4859    4863#
4866#   4869#   4873    4877#   4881    4884#   4888    4893#   4897    4919#   4924    4937#   4941
4959#   4963    4967#   4970#   4973#   4977    4981#   4985    5001#   5006    5025#   5029    5033#
5036#   5039#   5043    5047#   5051    5054#   5058    5063#   5067    5085#   5090    5106#   5110
5114#   5117#   5120#   5124    5128#   5132    5137#   5141    5178#   5182    5211#   5216    5229#
5233    5251#   5255    5259#   5262#   5265#   5269    5273#   5277    5300#   5305    5322#   5326
5330#   5333#   5336#   5340    5344#   5348    5351#   5355    5360#   5364    5379#   5384    5400#
5404    5408#   5411#   5414#   5418    5422#   5426    5429#   5433    5438#   5442    5464#   5469
5482#   5486    5504#   5508    5512#   5515#   5518#   5522    5526#   5530    5549#   5554    5560#
5564    5568#   5569#   5570#   5574    5576#   5580    5600#   5604    5608#   5611#   5615    5619#
5623    5626#   5630    5635#   5639    5657#   5662    5668#   5672    5676#   5677#   5678#   5682
5684#   5688    5707#   5711    5716#   5719#   5723    5727#   5731    5734#   5738    5743#   5747
5750#   5756    5780#   5785    5798#   5802    5820#   5824    5828#   5831#   5834#   5838    5842#
5846    5869#   5874    5891#   5895    5899#   5902#   5905#   5909    5913#   5917    5920#   [illegible]
5929#   5933    5948#   5953    5969#   5973    5977#   5980#   5983#   5987    5991#   5995    [illegible]
6002    6007#   6011    6033#   6038    6051#   6055    6073#   6077    6081#   6084#   6087#   [illegible]
6095#   6099    6122#   6127    6144#   6148    6152#   6155#   6158#   6162    6166#   6170    6173#
6177    6182#   6186    6201#   6206    6222#   6226    6230#   6233#   6236#   6240    6244#   6248
6251#   6255    6260#   6264    6290#   6295    6308#   6312    6337#   6341    6345#   6348#   6351#
6355    6359#   6363    6379#   6384    6403#   6407    6411#   6414#   6417#   6421    6425#   6429
6446#   6450    6465#   6469    6482#   6486    6494#   6498    6512#   6517    6535#   6539    6543#
6546#   6549#   6553    6557#   6561    6577#   6581    6596#   6600    6613#   6617    6625#   6629
6654#   6659    6699#   6704    6717#   6721    6746#   6750    6754#   6757#   6760#   6764    6768#
6772    6788#   6793    6812#   6816    6820#   6823#   6826#   6830    6834#   6838    6854#   6858
6863#   6867    6881#   6886    6903#   6907    6911#   6914#   6917#   6921    6925#   6929    6945#
6949    6954#   6958    6961#   6967    6990#   6995    7035#   7050#   7055    7092#   7096    7100#
7103#   7106#   7110    7114#   7118    7123#   7127    7137#   7141    7156#   7161    7197#   7201
7205#   7208#   7211#   7215    7219#   7223    7228#   7232    7242#   7246    7274#   7288#   7293
7322#   7323#   7327    7329#   7333    7343#   7346#   7350    7354#   7358    7362#   7366    7381#
7386    7413#   7415#   7419    7423#   7427    7438#   7441#   7445    7449#   7453    7456#   7460
7482#   7487    7500#   7504    7522#   7526    7530#   7533#   7536#   7540    7544#   7548    7564#
7569    7593#   7597    7601#   7604#   7607#   7611    7615#   7619    7622#   7626    7631#   7635
7650#   7655    7675#   7679    7683#   7686#   7689#   7693    7697#   7701    7704#   7708    7713#
7717    7720#   7726    7747#   7752    7765#   7769    7787#   7791    7795#   7798#   7801#   7805
7809#   7813    7829#   7834    7853#   7857    7861#   7864#   7867#   7871    7875#   7879    7882#
7886    7891#   7895    7910#   7915    7933#   7937    7941#   7944#   7947#   7951    7955#   7959
7962#   7966    7971#   7975    7978#   7984    8011#   8016    8029#   8033    8051#   8055    8059#
8062#   8065#   8069    8073#   8077    8093#   8098    8117#   8121    8125#   8128#   8131#   8135
8139#   8143    8146#   8150    8155#   8159    8174#   8179    8195#   8199    8203#   8206#   8209#
8213    8217#   8221    8224#   8228    8233#   8237    8259#   8264    8277#   8281    8299#   8303
8307#   8310#   8313#   8317    8321#   8325    8341#   8346    8365#   8369    8373#   8376#   8379#
8383    8387#   8391    8394#   8398    8403#   8407    8422#   8427    8444#   8448    8452#   8455#
8458#   8462    8466#   8470    8473#   8477    8482#   8486    8489#   8495    8516#   8521    8534#
8538    8556#   8560    8564#   8567#   8570#   8574    8578#   8582    8598#   8603    8622#   8626
8630#   8633#   8636#   8640    8644#   8648    8651#   8655    8660#   8664    8679#   8684    8701#
8705    8709#   8712#   8715#   8719    8723#   8727    8730#   8734    8739#   8743    8746#   8752
8773#   8778    8791#   8795    8813#   8817    8821#   8824#   8827#   8831    8835#   8839    8855#
8860    8879#   8883    8887#   8890#   8893#   8897    8901#   8905    8908#   8912    8917#   8921
8936#   8941    8958#   8962    8966#   8969#   8972#   8976    8980#   8984    8987#   8991    8996#
```

```
                    9000    9003#   9009
BADSCT  040734      3635    3892    4140    4432    4689    4937    5229    5482    5798    6051    6308    6717    7500
                    7765    8029    8277    8534    8791    9463#
BAI     = 000010    1177#   7578   10998
BB00    = 000001    1095#
BB01    = 000002    1094#
BB02    = 000004    1093#
BB03    = 000010    1092#
BB04    = 000020    1091#
BB05    = 000040    1090#
BB06    = 000100    1089#
BB07    = 000200    1088#
BB08    = 000400    1087#
BB09    = 001000    1086#
BIT0    = 000001     884#   3269    3590    3592   10285
BIT00   = 000001     874#    884     910     959     978     997    1035    1054    1095    1181    1197
BIT01   = 000002     873#    883     909     958     996    1034    1053    1094    1180    1196
BIT02   = 000004     872#    882     908     957     995    1033    1052    1093    1179    1195
BIT03   = 000010     871#    881     907     956     994    1032    1051    1092    1107    1177    1194
BIT04   = 000020     870#    880     906     955     993    1050    1091    1176
BIT05   = 000040     869#    879     905     992    1031    1049    1090    1175
BIT06   = 000100     868#    878     977     991    1013    1030    1048    1089    1160    1174    1193
BIT07   = 000200     867#    877     976     990    1012    1029    1047    1071    1088    1106    1159    1173
BIT08   = 000400     866#    876     954     975     989    1011    1028    1046    1087    1158    1171   13268
BIT09   = 001000     865#    875     953     974     988    1010    1027    1045    1086    1157    1170   13286   13378
BIT1    = 000002     883#   9753    9911   11015
BIT10   = 002000     864#    952     973     987    1009    1026    1044    1070    1085    1105    1156    1169    1192
                    13355
BIT11   = 004000     863#    904     972     986    1025    1043    1061    1069    1084    1104    1168    1191    9381
                    13293
BIT12   = 010000     862#    971     985    1024    1042    1068    1083    1103    1167    1190    9331
BIT13   = 020000     861#    970     984    1023    1041    1060    1082    1102    1154    1166    1189    6286    6640
                    13362
BIT14   = 040000     860#    969     983    1022    1040    1059    1081    1101    1112    1153    1165    1188    9296
                    13254
BIT15   = 100000     859#    968     982    1021    1039    1058    1080    1100    1111    1152    1164    1187    9283
                    9996   10050
BIT2    = 000004     882#   10285
BIT3    = 000010     881#   9409    12457   12544
BIT4    = 000020     880#   9359    12563
BIT5    = 000040     879#
BIT6    = 000100     878#   9308
BIT7    = 000200     877#   3264    10291
BIT8    = 000400     876#
BIT9    = 001000     875#
BOTADR  037756      9147*   9165*   9168    9183    9228#
BOTFLG  037760      9133*   9175*   9178    9181*   9229#
BPTVEC= 000014       891#
BSE     = 100000    1100#   6479    6490    6610    6621    6846    6849    6937    6940    9659   10902
BUFFER  107550      9478*   9727   13835#
BUFONE  107550      3628    3711    3848    3885    3968    4133    4217    4425    4508    4645    4682    4765    4930
                    5014    5222    5286    5475    5589    5751    5791    5855    6044    6109    6301    6319    6321*
                    6323*   6391    6648    6710    6728    6730*   6732*   6802    6962    6985    7029    7082    7268
                    7493    7581    7721    7758    7842    7979    8022    8106    8270    8354    8490    8527    8611
                    8747    8784    8867    9004   13835#
BUFTWO  110554      3791    3849    4285*   4351    4366    4368    4389*   4588    4646    5082*   5148    5163    5165
```

```
                    5186*  5697   5752   6523   6893   6963   7187   7664   7722   7921   7980   8434   8491
                    8690   8748   8947   9005  13835#
CC     = 004000     1084#
CH     = 002000     1085#
CHRCNT  037761      9134*  9152*  9158*  9159   9162*  9166   9171*  9182*  9230#
CKSWR  = 104410     13252  13351  13377  13728#
CLEAN  = ****** U    6284   6638  13835
CLKADR  001502      1441#  10247* 10249  10253* 10255  10280
CLKVCT  001504      1442#  10248* 10254*
CLR    = 000040     1175#   3271  11441
CLRSTS  053016      9318   11477#
CLSPRN  067576      3342    3358   3379  13835#
CMNSTA  006746      3294    3429#
CMPBUF  043104      3847    4644   5750   6961   7720   7978   8489   8746   9003   9976#
CMPERR  063142     12856#
CNSL00  070074      3304   13835#
CNSL01  070133      3339   13835#
CNSL02  070160      3348   13835#
CNSL03  070241      3353   13835#
CNSL04  070271      3364   13835#
CNSL05  070345      3374   13835#
CNSL06  070401      3385   13835#
CNSL07  070426      3398   13835#
CNTCLR  052700      3529    3573   9298  11433#
CONT   = 000100     1048#
CR     = 000015      799#   3412   9150  12938  13227  13237  13835
CRLF   = 000200      800#   3248  13198  13237
CTLFG   001326      1371#  3468*  12929  12943*
CYLMSK= 001777      1075#   9914
DBCK   = 100000     1021#
DBEN   = 040000     1022#
DBL    = 002000     1192#
DCK    = 100000      982#    999   9652   9984  10940  10960  12550  12553
DDISP  = 177570      806#   1292   3212
DEBL   = 020000     1023#
DEVSEL  051226      9285   11143#
DISPLA  001156      1292#   3212*  3220*  13308*  13354*
DISPRE  000174      1219#   3220
DLT    = 100000     1164#  10784  12629  12632
DMD    = 000001     1035#   1054#  7061   7167
DPE    = 000010     1107#   9653   9659  10446  10726  11222  11761  12182  12603  12606
DPEHI  = 040000     1188#
DPELO  = 020000     1189#
DPR    = 000400      975#  11383  11580  11581  12047
DRQ    = 004000     1061#
DRVCLR= 000010      919#   9506  12035
DRVSTS  056234      9524   12026#
DRY    = 000200      976#  10413  10509  10512  10513  11383  11580  11581  12047
DSWR   = 177570      805#   1291   3211
DTASTS  057036      3679   3752   3831   3936   4009   4087   4184   4257   4340   4476   4549   4628   4733
                    4806   4884   4981   5054   5137   5273   5351   5429   5526   5626   5734   5842   5920
                    5998   6095   6173   6251   6359   6446   6465   6482   6577   6596   6613   6768   6854
                    6945   7123   7228   7544   7622   7704   7809   7882   7962   8073   8146   8224   8321
                    8394   8473   8578   8651   8730   8835   8908   8987   9627  12157#
DTE    = 010000      985#    999  12416  12434  12437
DTO    = 010000     1024#
```

```
DULPRT= 024024          1064#   3588   3592   3595
DVA   = 004000           904#   3277   3548  10136  10209  10374  10377  11168  11486  12034  12035
DVC   = 000200          1106#  11315  11570  11822  11860  11863  11996  12369  12373  12376  12421  12795  12800
                       12803
EARLY   071322          8283  13835#
EBL   = 020000          1041#
ECH   = 000100           991#    999   9652   9986  10923  10938  10956  10962  12560  13835
ECI   = 004000          1069#   9988  10958  12557
ECRC  = 001000          1045#
EDT1    075652          1476   1483   1490   1497   1504   1511   1525   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   1771   1778   1785   1792   1799   1807   1815   1823   1837   1844   1851
                        1858   1865   1872   1879   1887   1894   1901   1908   1915   1922   1929   1936   1943
                        1950   1957   1964   1971   2013   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   2287   2294   2301   2308   2322   2329   2336   2343   2350   2357
                        2364   2371   2378   2385   2392   2399   2406   2413   2420   2427   2434   2441   2448
                        2469   2476   2483   2490   2497   2637   2644   2651   2672   2679   2686   2700   2707
                        2714   2721   2728   2735   2742   2749   2757   2764   2772   2779   2786   2794   2802
                        2810   2819   2827   2835   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   3066   3073   3087   3094   3101   3108   3115
                        3136  13835#
EDT110  075670          1978  13835#
EDT111  075672          1985   1992  13835#
EDT114  075674          2006  13835#
EDT2    075662          2455   2462   2658   2665  13835#
EDT223  075676          2504   2693  13835#
EDT336  075706          2280   3038   3052   3059  13835#
EDT337  075716          3045  13835#
EDT344  075726          3080  13835#
EDT353  075740          3129  13835#
ED1     105324         13835#
ED110   105332         13835#
ED111   105336         13835#
ED114   105344         13835#
ED223   105354         13835#
ED336   105364         13835#
ED337   105376         13835#
ED353   105410         13835#
EECC  = 000020          1050#
EFT1    075742          1477   1484   1491   1498   1505   1512   1526   1533   1540   1547   1554   1561   1568
                        1575   1582   1589   1596   1603   1610   1617   1624   1631   1638   1645   1652   1659
                        1666   1673   1680   1687   1694   1701   1708   1715   1722   1729   1736   1743   1750
                        1757   1764   1772   1779   1786   1793   1800   1808   1816   1824   1838   1845   1852
                        1859   1866   1873   1880   1888   1895   1902   1909   1916   1923   1930   1937   1944
                        1951   1958   1965   1972   2014   2021   2028   2035   2042   2049   2056   2063   2070
                        2077   2084   2091   2098   2105   2112   2119   2126   2133   2140   2147   2154   2161
                        2168   2175   2182   2189   2196   2203   2210   2217   2224   2231   2238   2245   2252
                        2259   2266   2273   2288   2295   2302   2309   2323   2330   2337   2344   2351   2358
                        2365   2372   2379   2386   2393   2400   2407   2414   2421   2428   2435   2442   2449
                        2470   2477   2484   2491   2498   2638   2645   2652   2673   2680   2687   2701   2708
                        2715   2722   2729   2736   2743   2750   2758   2765   2773   2780   2787   2795   2803
                        2811   2820   2828   2836   2843   2850   2857   2864   2871   2878   2885   2892   2899
```

```
                    2906    2913    2920    2927    2934    2941    2948    2955    2962    2969    2976    2983    2990
                    2997    3004    3011    3018    3025    3032    3067    3074    3088    3095    3102    3109    3116
                    3137   13835#
EFT110   075760     1979   13835#
EFT111   075762     1986    1993   13835#
EFT114   075764     2007   13835#
EFT2     075752     2456    2463    2659    2666   13835#
EFT223   075766     2505    2694   13835#
EFT336   075776     2281    3039    3053    3060   13835#
EFT337   076006     3046   13835#
EFT344   076016     3081   13835#
EFT353   076026     3130   13835#
EF110    105502    13835#
EF111    105503    13835#
EF114    105505    13835#
EF336    105510    13835#
EF337    105514    13835#
EHT1     075526     1475    1482    1489    1496    1503    1510    1524    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    1770    1777    1784    1791    1798    1806    1814    1822    1836    1843    1850
                    1857    1864    1871    1878    1886    1893    1900    1907    1914    1921    1928    1935    1942
                    1949    1956    1963    1970    2012    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    2286    2293    2300    2307    2321    2328    2335    2342    2349    2356
                    2363    2370    2377    2384    2391    2398    2405    2412    2419    2426    2433    2440    2447
                    2468    2475    2482    2489    2496    2636    2643    2650    2671    2678    2685    2699    2706
                    2713    2720    2727    2734    2741    2748    2756    2763    2771    2778    2785    2793    2801
                    2809    2818    2826    2834    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    3065    3072    3086    3093    3100    3107    3114
                    3135   13835#
EHT110   075550     1977   13835#
EHT111   075554     1984    1991   13835#
EHT114   075560     2005   13835#
EHT2     075540     2454    2461    2657    2664   13835#
EHT223   075564     2503   13835#
EHT256   075576     2692   13835#
EHT336   075610     2279    3037    3051    3058   13835#
EHT337   075622     3044   13835#
EHT344   075634     3079   13835#
EHT353   075646     3128   13835#
EH1      104257    13835#
EH110    104276    13835#
EH111    104305    13835#
EH114    104324    13835#
EH223    104353    13835#
EH256    104401    13835#
EH336    104456    13835#
EH337    104515    13835#
EH344    104654    13835#
EH353    105014    13835#
EMS1     076030    13835#
EMS10    076364    13835#
EMS100   100604    13835#
```

```
EMS101  100632          13835#
EMS102  100660          13835#
EMS103  100707          13835#
EMS104  100716          13835#
EMS105  100745          13835#
EMS106  100774          13835#
EMS11   076426          13835#
EMS110  101013          13835#
EMS111  101042          13835#
EMS112  101053          13835#
EMS113  101137          13835#
EMS114  101167          13835#
EMS115  101213          13835#
EMS116  101220          13835#
EMS117  101250          13835#
EMS12   076437          13835#
EMS120  101272          13835#
EMS121  101307          13835#
EMS122  101352          13835#
EMS123  101407          13835#
EMS124  101452          13835#
EMS125  101504          13835#
EMS126  101547          13835#
EMS127  101612          13835#
EMS13   076500          13835#
EMS130  101651          13835#
EMS131  101707          13835#
EMS132  101747          13835#
EMS133  101773          13835#
EMS134  102016          13835#
EMS135  102060          13835#
EMS136  102122          13835#
EMS137  102147          13835#
EMS14   076511          13835#
EMS140  102204          13835#
EMS141  102225          13835#
EMS142  102253          13835#
EMS143  102270          13835#
EMS144  102316          13835#
EMS145  102326          13835#
EMS146  102355          13835#
EMS147  102374          13835#
EMS15   076522          13835#
EMS150  102411          13835#
EMS151  102430          13835#
EMS152  102454          13835#
EMS153  102500          13835#
EMS154  102522          13835#
EMS155  102536          13835#
EMS156  102556          13835#
EMS157  102574          13835#
EMS16   076544          13835#
EMS160  102611          13835#
EMS161  102626          13835#
EMS162  102656          13835#
EMS163  102711          13835#
```

EMS164	102735	13835#
EMS165	103016	13835#
EMS166	103032	13835#
EMS167	103057	13835#
EMS17	076574	13835#
EMS170	103104	13835#
EMS171	103133	13835#
EMS172	103166	13835#
EMS173	103206	13835#
EMS174	103223	13835#
EMS175	103231	13835#
EMS176	103257	13835#
EMS177	103306	13835#
EMS2	076077	13835#
EMS20	076634	13835#
EMS200	103330	13835#
EMS201	103345	13835#
EMS202	103413	13835#
EMS203	103443	13835#
EMS204	103456	13835#
EMS205	103504	13835#
EMS206	103516	13835#
EMS207	103543	13835#
EMS21	076657	13835#
EMS210	103570	13835#
EMS211	103601	13835#
EMS212	103617	13835#
EMS213	103642	13835#
EMS214	103665	13835#
EMS215	103703	13835#
EMS216	103724	13835#
EMS217	103754	13835#
EMS22	076702	13835#
EMS220	104007	13835#
EMS221	104027	13835#
EMS222	104077	13835#
EMS223	104130	13835#
EMS224	104205	13835#
EMS23	076716	13835#
EMS24	076744	13835#
EMS25	076773	13835#
EMS26	077010	13835#
EMS27	077031	13835#
EMS3	076114	13835#
EMS30	077042	13835#
EMS31	077052	13835#
EMS32	077101	13835#
EMS33	077130	13835#
EMS34	077156	13835#
EMS35	077227	13835#
EMS36	077256	13835#
EMS37	077305	13835#
EMS4	076157	13835#
EMS40	077333	13835#
EMS41	077362	13835#
EMS42	077410	13835#

```
EMS43   077437          13835#
EMS44   077466          13835#
EMS45   077541          13835#
EMS46   077604          13835#
EMS47   077633          13835#
EMS5    076222          13835#
EMS50   077662          13835#
EMS51   077711          13835#
EMS52   077737          13835#
EMS53   077751          13835#
EMS54   077763          13835#
EMS55   100005          13835#
EMS56   100034          13835#
EMS57   100111          13835#
EMS6    076252          13835#
EMS60   100137          13835#
EMS61   100152          13835#
EMS62   100201          13835#
EMS63   100217          13835#
EMS64   100245          13835#
EMS65   100263          13835#
EMS66   100331          13835#
EMS67   100401          13835#
EMS7    076317          13835#
EMS70   100426          13835#
EMS71   100440          13835#
EMS72   100453          13835#
EMS73   100463          13835#
EMS74   100500          13835#
EMS75   100517          13835#
EMS76   100525          13835#
EMS77   100556          13835#
EMTVEC= 000030            894#   3195#   3196#
EMT1    071434           1474  13835#
EMT10   071504           1523  13835#
EMT100  072502           1920  13835#
EMT101  072520           1927  13835#
EMT102  072536           1934  13835#
EMT103  072546           1941  13835#
EMT104  072560           1948  13835#
EMT105  072576           1955  13835#
EMT106  072606           1962  13835#
EMT107  072616           1969  13835#
EMT11   071510           1530  13835#
EMT110  072636           1976  13835#
EMT111  072650           1983  13835#
EMT112  072656           1990  13835#
EMT113  072664           1997  13835#
EMT114  072700           2004  13835#
EMT115  072706           2011  13835#
EMT116  072716           2018  13835#
EMT117  072726           2025  13835#
EMT12   071514           1537  13835#
EMT120  072736           2032  13835#
EMT121  072746           2039  13835#
EMT122  072756           2046  13835#
```

EMT123	072766	2053	13835#
EMT124	072776	2060	13835#
EMT125	073006	2067	13835#
EMT126	073016	2074	13835#
EMT127	073030	2081	13835#
EMT13	071522	1544	13835#
EMT130	073042	2088	13835#
EMT131	073054	2095	13835#
EMT132	073066	2102	13835#
EMT133	073100	2109	13835#
EMT134	073112	2116	13835#
EMT135	073124	2123	13835#
EMT136	073136	2130	13835#
EMT137	073150	2137	13835#
EMT14	071534	1551	13835#
EMT140	073160	2144	13835#
EMT141	073170	2151	13835#
EMT142	073200	2158	13835#
EMT143	073210	2165	13835#
EMT144	073220	2172	13835#
EMT145	073230	2179	13835#
EMT146	073240	2186	13835#
EMT147	073250	2193	13835#
EMT15	071542	1558	13835#
EMT150	073260	2200	13835#
EMT151	073270	2207	13835#
EMT152	073302	2214	13835#
EMT153	073314	2221	13835#
EMT154	073332	2228	13835#
EMT155	073350	2235	13835#
EMT156	073362	2242	13835#
EMT157	073374	2249	13835#
EMT16	071550	1565	13835#
EMT160	073406	2256	13835#
EMT161	073416	2263	13835#
EMT162	073430	2270	13835#
EMT163	073442	13835#	
EMT164	073452	2285	13835#
EMT165	073460	2292	13835#
EMT166	073466	2299	13835#
EMT167	073474	2306	13835#
EMT17	071560	1572	13835#
EMT170	073502	13835#	
EMT171	073512	2320	13835#
EMT172	073522	2327	13835#
EMT173	073532	2334	13835#
EMT174	073542	2341	13835#
EMT175	073554	2348	13835#
EMT176	073562	2355	13835#
EMT177	073570	2362	13835#
EMT2	071440	1481	13835#
EMT20	071570	1579	13835#
EMT200	073602	2369	13835#
EMT201	073614	2376	13835#
EMT202	073626	2383	13835#
EMT203	073640	2390	13835#

EMT204	073652	2397	13835#
EMT205	073670	2404	13835#
EMT206	073700	2411	13835#
EMT207	073710	2418	13835#
EMT21	071600	1586	13835#
EMT210	073722	2425	13835#
EMT211	073730	2432	13835#
EMT212	073744	2439	13835#
EMT213	073760	2446	13835#
EMT214	073770	2453	13835#
EMT215	074010	2460	13835#
EMT216	074022	2467	13835#
EMT217	074032	2474	13835#
EMT22	071610	1593	13835#
EMT220	074042	2481	13835#
EMT221	074052	2488	13835#
EMT222	074062	2495	13835#
EMT223	074072	2502	13835#
EMT224	074102	13835#	
EMT225	074104	13835#	
EMT226	074106	13835#	
EMT227	074110	13835#	
EMT23	071620	1600	13835#
EMT230	074112	13835#	
EMT231	074114	13835#	
EMT232	074116	13835#	
EMT233	074120	13835#	
EMT234	074122	13835#	
EMT235	074124	13835#	
EMT236	074126	13835#	
EMT237	074130	13835#	
EMT24	071630	1607	13835#
EMT240	074132	13835#	
EMT241	074134	13835#	
EMT242	074136	13835#	
EMT243	074140	13835#	
EMT244	074142	13835#	
EMT245	074144	13835#	
EMT246	074146	2635	13835#
EMT247	074156	2642	13835#
EMT25	071640	1614	13835#
EMT250	074170	2649	13835#
EMT251	074204	2656	13835#
EMT252	074212	2663	13835#
EMT253	074220	2670	13835#
EMT254	074236	2677	13835#
EMT255	074246	2684	13835#
EMT256	074256	2691	13835#
EMT257	074266	2698	13835#
EMT26	071650	1621	13835#
EMT260	074300	2705	13835#
EMT261	074312	2712	13835#
EMT262	074332	2719	13835#
EMT263	074342	2726	13835#
EMT264	074352	2733	13835#
EMT265	074372	2740	13835#

EMT266	074412	2747	13835#	
EMT267	074424	2755	13835#	
EMT27	071660	1628	13835#	
EMT270	074442	2762	13835#	
EMT271	074456	2770	13835#	
EMT272	074474	2777	13835#	
EMT273	074512	2784	13835#	
EMT274	074530	2792	13835#	
EMT275	074546	2800	13835#	
EMT276	074560	2808	13835#	
EMT277	074574	2817	13835#	
EMT3	071446	1488	13835#	
EMT30	071670	1635	13835#	
EMT300	074612	2825	13835#	
EMT301	074632	2833	13835#	
EMT302	074654	2840	13835#	
EMT303	074666	2847	13835#	
EMT304	074700	2854	13835#	
EMT305	074712	2861	13835#	
EMT306	074724	2868	13835#	
EMT307	074734	2875	13835#	
EMT31	071700	1642	13835#	
EMT310	074754	2882	13835#	
EMT311	074766	2889	13835#	
EMT312	075000	2896	13835#	
EMT313	075012	2903	13835#	
EMT314	075032	2910	13835#	
EMT315	075044	2917	13835#	
EMT316	075056	2924	13835#	
EMT317	075070	2931	13835#	
EMT32	071710	1649	13835#	
EMT320	075100	2938	13835#	
EMT321	075110	2945	13835#	
EMT322	075120	2952	13835#	
EMT323	075126	2959	13835#	
EMT324	075136	2966	13835#	
EMT325	075150	2973	13835#	
EMT326	075162	2980	13835#	
EMT327	075200	2987	13835#	
EMT33	071720	1656	13835#	
EMT330	075212	2994	13835#	
EMT331	075222	3001	13835#	
EMT332	075234	3008	13835#	
EMT333	075246	3015	13835#	
EMT334	075264	3022	13835#	
EMT335	075302	3029	13835#	
EMT336	075322	2278	3036	13835#
EMT337	075332	3043	13835#	
EMT34	071730	1663	13835#	
EMT340	075342	3050	13835#	
EMT341	075354	3057	13835#	
EMT342	075362	3064	13835#	
EMT343	075374	3071	13835#	
EMT344	075406	3078	13835#	
EMT345	075420	3085	13835#	
EMT346	075430	3092	13835#	

```
EMT347  075444            3099   13835#
EMT35   071740            1670   13835#
EMT350  075456            3106   13835#
EMT351  075474            3113   13835#
EMT352  075504            3120   13835#
EMT353  075510            3127   13835#
EMT354  075520            3134   13835#
EMT36   071750            1677   13835#
EMT37   071760            1684   13835#
EMT4    071454            1495   13835#
EMT40   071770            1691   13835#
EMT41   071776            1698   13835#
EMT42   072006            1705   13835#
EMT43   072020            1712   13835#
EMT44   072030            1719   13835#
EMT45   072040            1726   13835#
EMT46   072050            1733   13835#
EMT47   072060            1740   13835#
EMT5    071462            1502   13835#
EMT50   072066            1747   13835#
EMT51   072076            1754   13835#
EMT52   072110            1761   13835#
EMT53   072126            1769   13835#
EMT54   072144            1776   13835#
EMT55   072154            1783   13835#
EMT56   072166            1790   13835#
EMT57   072204            1797   13835#
EMT6    071470            1509   13835#
EMT60   072214            1805   13835#
EMT61   072232            1813   13835#
EMT62   072252            1821   13835#
EMT63   072266          13835#
EMT64   072270            1835   13835#
EMT65   072310            1842   13835#
EMT66   072330            1849   13835#
EMT67   072342            1856   13835#
EMT7    071476            1516   13835#
EMT70   072354            1863   13835#
EMT71   072364            1870   13835#
EMT72   072374            1877   13835#
EMT73   072412            1885   13835#
EMT74   072430            1892   13835#
EMT75   072440            1899   13835#
EMT76   072460            1906   13835#
EMT77   072470            1913   13835#
ENRGDT  071434          13835#
EQUALS  067577          13835#
ERR     = 040000            969#  10555   10557   10573   10607
ERRNMB  037754           9114#   9115#   9118    9130    9135    9227#
ERRTYP  037216           9093#  13364
ERRVEC= 000004            887#   3209    3210#   3221#   3489    3490    3491#   3492#   3503#   3504#   3508#   3509#   3535
                         3536    3537#   3538#   3542#   3543#   3554#   3555#   10127   10128   10132#  10133#  10164#  10165#
                        10200   10201   10205#  10206#  10231#  10232#  10243   10244   10245#  10246#  10252#  10260#  10261#
                        10275   10276   10277#  10278#  10302#  10303#  11150   11151   11154#  11155#  11188#  11189#  11436
                        11437   11438#  11439#  11450#  11451#  13259   13260#  13262#  13265#
ERTY00  037762            9100    9232#
```

ERTY01 037770 9108 9233#
ERTY02 040000 9117 9235#
ERTY03 040007 9123 9237#
ERTY04 040015 9219 9239#
ESRC = 004000 1043#
FER = 000020 993# 999 6462 6472 6593 6603 6841 6844 6932 6935 9652 10887 12485
FIND = 000001 12502 12505 12528 12531

```
                    7883    7891#   7892    7910#   7912#   7933#   7934    7941#   7944#   7947#   7948    7955#   7956
                    7962#   7963    7971#   7972    7978#   7981    8011#   8013#   8029#   8030    8051#   8052    8059#
                    8062#   8065#   8066    8073#   8074    8093#   8095#   8117#   8118    8125#   8128#   8131#   8132
                    8139#   8140    8146#   8147    8155#   8156    8174#   8176#   8195#   8196    8203#   8206#   8209#
                    8210    8217#   8218    8224#   8225    8233#   8234    8259#   8261#   8277#   8278    8299#   8300
                    8307#   8310#   8313#   8314    8321#   8322    8341#   8343#   8365#   8366    8373#   8376#   8379#
                    8380    8387#   8388    8394#   8395    8403#   8404    8422#   8424#   8444#   8445    8452#   8455#
                    8458#   8459    8466#   8467    8473#   8474    8482#   8483    8489#   8492    8516#   8518#   8534#
                    8535    8556#   8557    8564#   8567#   8570#   8571    8578#   8579    8598#   8600#   8622#   8623
                    8630#   8633#   8636#   8637    8644#   8645    8651#   8652    8660#   8661    8679#   8681#   8701#
                    8702    8709#   8712#   8715#   8716    8723#   8724    8730#   8731    8739#   8740    8746#   8749
                    8773#   8775#   8791#   8792    8813#   8814    8821#   8824#   8827#   8828    8835#   8836    8855#
                    8857#   8879#   8880    8887#   8890#   8893#   8894    8901#   8902    8908#   8909    8917#   8918
                    8936#   8938#   8958#   8959    8966#   8969#   8972#   8973    8980#   8981    8987#   8988    8996#
                    8997    9003#   9006
FMT16 = 010000       1068#   3630    3887    4135    4427    4684    4932    5224    5477    5793    6046    6303    6436
                    6567    6650    6712    6800    6987    7031    7270    7495    7760    8024    8272    8529    8786
                    9488    9917    9919    9990    11247   12323
FNCDTB  070676      10601   13835#
FNCMSK= 000077        911#  12034   12478
F0      = 000002      909#  10500
F1      = 000004      908#  10500
F2      = 000010      907#  10500   11074
F3      = 000020      906#  10500
F4      = 000040      905#  10500
GENBUF  042640      3643    3900    4148    4440    4697    4945    5237    5290    5490    5806    5859    6059    6112
                    6316    6652    6725    6989    7508    7773    8037    8285    8542    8799    9901#
GET     043536      3582    3671    3737    3816    3928    3994    4072    4176    4242    4323    4358    4468    4534
                    4613    4725    4791    4869    4973    5039    5120    5156    5265    5336    5414    5518    5570
                    5611    5678    5719    5834    5905    5983    6087    6158    6236    6351    6417    6549    6760
                    6826    6917    7106    7211    7329    7346    7423    7441    7536    7607    7689    7801    7867
                    7947    8065    8131    8209    8313    8379    8458    8570    8636    8715    8827    8893    8972
                    9311    9335    9362    9385    9411    9515    9548    9584    9618   10118#
GETBUF  001330      1376#  10130
GETINX  001506      1447#   3579#   3580#   4356#   4357#   5153#   5154#   9333#   9334#   9383#   9384#  10076   10131
GETSTS  043452      3665    3731    3810    3922    3988    4066    4170    4236    4317    4462    4528    4607    4719
                    4785    4863    4967    5033    5114    5259    5330    5408    5512    5568    5676    5828    5899
                    5977    6081    6152    6230    6345    6411    6543    6754    6820    6911    7035    7100    7205
                    7274    7322    7413    7530    7601    7683    7795    7861    7941    8059    8125    8203    8307
                    8373    8452    8564    8630    8709    8821    8887    8966    9310    9361    9404    9502   10072#
GNS     = ****** U   1218    3247    9052    9059   13719   13720   13721   13722   13723   13724   13726   13728   13729
                   13730   13731   13732   13733
GO      = 000001      910#  3631    3713    3792    3888    3970    4048    4136    4218    4299    4428    4510    4589
                    4685    4767    4845    4933    5015    5096    5225    5312    5390    5478    5559    5590    5667
                    5696    5794    5881    5959    6047    6134    6212    6304    6393    6525    6661    6713    6801
                    6892    6997    7081    7186    7271    7310    7402    7496    7582    7665    7761    7843    7923
                    8025    8107    8185    8273    8355    8433    8530    8612    8691    8787    8869    8948    9345
                    9394    9506    9539    9573    9607   10288   10399   10411   10416   10432   10522   10525   12479
GTSWR = 104407       3242    3463   13726#
HCE     = 000200      990#   999    6443    6454    6574    6585    6841    6844    6932    6935    9652   10850   10855
                   10870   10872   10885   10900   12485   12515   12518   12528   12531   13835
HCI     = 002000     1070#  6800   10848   12481
HCRC    = 000400      989#   999    9652   10857   12485   12489   12492   12528   12531
HELP    107550       3324   13835#
HELPOS  067607       3318   13835#
HT      = 000011      797#   9156   13196   13237
```

```
IAE    = 002000            987#  10754   11051   11066   11092   11238   11255   11754   11808   11811   12312   12327   13835
IDXMSK= 000077           1146#  10144   10147   10217   10220
IE     = 000100           1160#   1193#
ILF    = 000001            997#  10626   10628   11648   11708   11712   11754   11775   11778   12268   12285   12288   13835
ILF02  = 000002            916#
ILF24  = 000024            926#
ILF26  = 000026            927#
ILF30  = 000030            929#
ILF32  = 000032            930#
ILF34  = 000034            931#
ILF36  = 000036            932#
ILF40  = 000040            933#
ILF42  = 000042            934#
ILF44  = 000044            935#
ILF46  = 000046            936#
ILF54  = 000054            939#
ILF56  = 000056            940#
ILF64  = 000064            943#
ILF66  = 000066            944#
ILF74  = 000074            947#
ILF76  = 000076            948#
ILR    = 000002            996#  11648   11694   11698   11962   11966   11969   12268   12271   12274
ILRG50= 000050           1132#
ILRG52= 000052           1133#
ILRG54= 000054           1134#
ILRG56= 000056           1135#
ILRG60= 000060           1136#
ILRG62= 000062           1137#
ILRG64= 000064           1138#
ILRG66= 000066           1139#
ILRG70= 000070           1140#
ILRG72= 000072           1141#
ILRG74= 000074           1142#
ILRG76= 000076           1143#
IOTVEC= 000020            892#   3193*   3194*
IPCK0  = 000001          1197#
IPCK1  = 000002          1196#
IPCK2  = 000004          1195#
IPCK3  = 000010          1194#
IR     = 000100          1174#  11511
IVC    = 010000          1103#   7121    7133    7226    7236   10669   10670   11273   11396   11822   11829   11832   11901
                        12251   12255   12683   12764   13835
LBC    = 002000          1105#  12369   12387   12392
LBT    = 002000           973#  10829   11053   11055
LF     = 000012           798#   9154   12938   13231   13237   13835
LS     = 000004          1052#  11535   12087
LSC    = 004000          1104#  12369   12403   12406
LST    = 000002          1053#  11535   12087
MCLK   = 004000          1025#
MCPE   = 020000          1154#  10399   10416   10429   10432   12166   12169
MDF    = 000100          1030#
MDPE   = 000400          1171#  10925   12575   12578
MEDENB  001474           1435#   3475*   9344*   9467    9723*
MFGFIL  105530           9489    9692    9705*   9712*   9773   13835#
MI     = 000004          1033#
MIXED   071006           8540    8797   13835#
```

```
MOC     = 000400          1028#
MOH     = 020000          1060#
MOL     = 010000           971#  9644  11334  11344  11345  11350  11355  11383  11634  11637  11798  11875  11876
                          11883  11888  12047  12228  12389  12642  12643  12665  12670  12741  12742  12746  12751
MRD     = 002000          1026#
MS      = 000040          1031#
MSC     = 000002          1034#
MSE     = 100000          1111#  6438  6569  9692  9915
MSER    = 000200          1029#
MUR     = 001000          1027#
MWD     = 000010          1051#  11536  12088
MWP     = 000010          1032#
MXF     = 001000          1170#  10814  12189  12203  12206
NDTMSK= 115760             999#
NED     = 010000          1167#  3274  3544  10381  10385  11163
NEM     = 004000          1168#  10799
NOP     = 000000           915#
NOTEX     070652          3282  13835#
NSA     = 100000          1058#
OCC     = 100000          1039#
OFD     = 000200          1071#
OFFSET= 000014             921#
OM      = 000001           978#  11875  11931  11934  12046  12590  12593
ONES      071046          4438  4695  4943  5288  5488  5804  6110  6723  7506  8035  13835#
OPE     = 020000          1102#
OPI     = 020000           984#  10653  11328  11352  11648  11666  11670  11754  11791  11794  11885  12223  12226
                          12667  12748  13835
OR      = 000200          1173#
PACACK= 000022             925#  9539
PAKACK= 000022             924#  925  9345
PAR     = 000010           994#  9655  10444  10448  10460  11220  11754  11759  11764  12180  12185
PAT     = 000020          1176#
PDA     = 000400          1046#
PGE     = 002000          1169#
PGM     = 001000           974#  12046
PHA     = 000200          1047#
PIP     = 020000           970#  7334  7337  7428  7431  9392  9570  11344  11365  11370  11875  11946  11951
                          12046  12642  12648  12653  12741  12778  12783
PIRQ    = 177772           804#
PIRQVE= 000240             898#
PLFS    = 002000          1044#
PRIERR    044532          3745  3824  4002  4080  4250  4331  4542  4621  4799  4877  5047  5128  5344
                          5422  5619  5727  5913  5991  6166  6244  6425  6557  6834  6925  7114  7219
                          7354  7449  7615  7697  7875  7955  8139  8217  8387  8466  8644  8723  8901
                          8980  10347#
PROMPT    067601          3399  3408  13835#
PR0     = 000000           821#
PR1     = 000040           822#
PR2     = 000100           823#
PR3     = 000140           824#  12914
PR4     = 000200           825#
PR5     = 000240           826#
PR6     = 000300           827#  3180  3470  3492  3538  10133  10206  10246  10278  11155  11439  12920
PR7     = 000340           828#
PS      = 177776           801#  802
PSEL    = 002000          1156#
```

```
PSW    = 177776            802#
PUT      044006           3657   3723   3802   3914   3980   4058   4162   4228   4309   4454   4520   4599   4711
                          4777   4855   4959   5025   5106   5251   5322   5400   5504   5560   5600   5668   5707
                          5820   5891   5969   6073   6144   6222   6337   6403   6535   6670   6746   6812   6903
                          7006   7062   7071   7092   7168   7177   7197   7300   7323   7392   7415   7522   7593
                          7675   7787   7853   7933   8051   8117   8195   8299   8365   8444   8556   8622   8701
                          8813   8879   8958   9348   9397   9507   9540   9574   9608  10194#
PUTBUF   001400          1403#  9478   9727#  10203
PUTINX   001535          1452#  3646   3714   3793   3903   3971   4049   4151   4219   4300   4443   4511   4590
                          4700   4768   4846   4948   5016   5097   5240   5313   5391   5493   5591   5698   5809
                          5882   5960   6062   6135   6213   6326   6394   6526   6662   6735   6803   6894   6998
                          7059*  7060*  7068*  7069*  7083   7165*  7166*  7174*  7175*  7188   7297*  7298*  7311
                          7389*  7390*  7403   7511   7583   7666   7776   7844   7924   8040   8108   8186   8288
                          8356   8435   8545   8613   8692   8802   8870   8949   9346*  9347*  9395*  9396*  9491
                         10204  10451
PWRVEC=  000024            893#  3199*  3200*  13738*  13739*  13748*  13754*  13766*  13767*
QSTMRK   067605           3313   3418  13835#
RCLSTS   054472           9418   9593  11745#
RD     = 000070            945#  3792   4589   5696   6525   6892   7186   7402   7665   7923   8433   8691   8948
RDCHR  = 104411           3305   3319   3329   3332   3400   3409  13591  13729#
RDLIN  = 104412          13661  13730#
RDOCT  = 104413           3343   3359   3380  13731#
RDY    = 000200           1159#  10288  10289  10395  10398  11486
READY    007100           3310   3467#  9027   12931
RECAL  = 000006            918#  9394   9573
RESREG = 104415           9223  13733#
RESVEC = 000010            888#
REX    = 010000           1042#
RG     = 040000           1040#
RGDTPT   071006          13835#
RH     = 000072            946#  9607
RIP    = 000020            923#
RLEASE = 000012            920#
RMAS   = 000016           1120#
RMASI    001346           1386#  12067  13835
RMASO    001416           1413#
RMBA   = 000004           1202#  3497*  3498   3649   3718   3797   3906   3975   4053   4154   4223   4304   4446
                          4515   4594   4703   4772   4850   4951   5020   5101   5243   5317   5395   5496   5595
                          5702   5812   5886   5964   6065   6139   6217   6329   6398   6530   6666   6738   6807
                          6898   7002   7087   7192   7315   7407   7514   7588   7670   7779   7848   7928   8043
                          8112   8190   8291   8360   8439   8548   8617   8696   8805   8874   8953   9496
RMBAE  = 000050           1205#
RMBAI    001334           1381#  4369   5166  11001  11003  11497  13835
RMBAO    001404           1406#  3628*  3711*  3791*  3885*  3968*  4133*  4217*  4425*  4508*  4588*  4682*  4765*
                          4930*  5014*  5222*  5286*  5475*  5589*  5697*  5791*  5855*  6044*  6109*  6301*  6391*
                          6523*  6648*  6710*  6802*  6893*  6985*  7029*  7082*  7187*  7268*  7493*  7581*  7664*
                          7758*  7842*  7921*  8022*  8106*  8270*  8354*  8434*  8527*  8611*  8690*  8784*  8867*
                          8947*  9489*  9907   9994  10997  11000
RMCC   = 000036           1127#
RMCCI    001366           1394#
RMCCO    001436           1421#
RMCS1  = 000000           1116#  1200#  3276   3494*  3540   3652   3720   3799   3909   3977   4055   4157   4225
                          4306   4449   4517   4596   4706   4774   4852   4954   5022   5103   5246   5319   5397
                          5499   5597   5704   5815   5888   5966   6068   6141   6219   6332   6400   6532   6668
                          6741   6809   6900   7004   7089   7194   7317   7409   7517   7590   7672   7782   7850
                          7930   8046   8114   8192   8294   8362   8441   8551   8619   8698   8808   8876   8955
```

```
                    9346   9395   9497  10135  10208  10287  11161
RMCS1I  001330      1379#  10374  10376  10378  10395  10397  10400  10411  10415  10417  10429  10431  10433
                   10522  10524  10535  10578  10984  11484  12033  12166  12168  12170  12189  13835
RMCS10  001400      1406#   3631*  3713*  3792*  3888*  3970*  4048*  4136*  4218*  4299*  4428*  4510*  4589*
                    4685*  4767*  4845*  4933*  5015*  5096*  5225*  5287*  5312*  5390*  5478*  5559*  5590*
                    5667*  5696*  5794*  5858*  5881*  5959*  6047*  6107*  6134*  6212*  6304*  6393*  6525*
                    6651*  6661*  6713*  6801*  6892*  6988*  6997*  7032*  7081*  7186*  7271*  7310*  7402*
                    7496*  7582*  7665*  7761*  7843*  7923*  8025*  8107*  8185*  8273*  8355*  8433*  8530*
                    8612*  8691*  8787*  8869*  8948*  9345*  9394*  9506*  9539*  9573*  9607*  9753   9911
                   10499  11015  12457  12477
RMCS2 = 000010       1203#   3271*  3272*  3273   3499*  3500   3541   7584  10134  10207  11160*  11162  11441*
                   11443*
RMCS2I  001340      1383#   4338   4347   4349   5135   5144   5146  10356  10381  10383  10384  10570  10641
                   10742  10785  10800  10815  10926  10998  11506  12203  12205  12207  12575  12577  12579
                   12629  12631  12633  13835
RMCS20  001410      1410#   7579*  7663*
RMCS3 = 000052      1206#
RMDA  = 000006      1117#   3648   3715   3794   3905   3972   4050   4153   4220   4301   4445   4512   4591
                    4702   4769   4847   4950   5017   5098   5242   5314   5392   5495   5592   5699   5811
                    5883   5961   6064   6136   6214   6328   6395   6527   6663   6737   6804   6895   6999
                    7084   7189   7312   7404   7513   7585   7667   7778   7845   7925   8042   8109   8187
                    8290   8357   8436   8547   8614   8693   8804   8871   8950   9492
RMDAI   001336      1382#  11097  13835
RMDAO   001406      1409#   3627*  3884*  4132*  4424*  4681*  4929*  5221*  5474*  5790*  6043*  6300*  6647*
                    6709*  6984*  7028*  7267*  7492*  7757*  8021*  8269*  8526*  8783*  9011*  9012   9485*
                    9666*  9667   9669   9687   9695*  9747   9874*  9910  11023  11024  11243  11245  11249
                   12317  12319  12321
RMDB  = 000022      1204#   3501*  3502   4356   5153  10083
RMDBI   001352      1388#   4365   5155*  5162
RMDBO   001422      1415#
RMDC  = 000034      1126#   3647   3716   3796   3904   3973   4052   4152   4221   4303   4444   4513   4593
                    4701   4770   4849   4949   5018   5100   5241   5315   5394   5494   5593   5701   5810
                    5884   5963   6063   6137   6216   6327   6396   6529   6665   6736   6805   6897   7001
                    7085   7191   7313   7406   7512   7586   7669   7777   7846   7927   8041   8110   8189
                    8289   8358   8438   8546   8615   8695   8803   8872   8952   9493
RMDCI   001364      1393#  11109  13835
RMDCO   001434      1420#   3626*  3883*  4131*  4423*  4680*  4928*  5220*  5473*  5537   5541*  5557   5558*
                    5582*  5646   5649*  5665   5666*  5690*  5789*  6042*  6299*  6646*  6708*  6983*  7027*
                    7266*  7491*  7756*  7986*  7987   8020*  8268*  8525*  8782*  9486*  9746   9873*  9909
                   11025  11240  12314
RMDS  = 000012      1118#   9333   9383
RMDSI   001342      1384#   7334   7336   7338   7428   7430   7432   9342   9392   9537   9570   9644  10413
                   10509  10511  10556  10573  10607  10611  10667  10694  10829  11054  11279  11334  11343
                   11350  11354  11356  11365  11369  11371  11380  11382  11384  11394  11398  11400  11579
                   11620  11622  11624  11634  11636  11638  11798  11836  11874  11883  11887  11889  11899
                   11903  11904  11915  11917  11919  11931  11933  11935  11946  11950  11952  12045  12228
                   12230  12259  12389  12455  12590  12592  12594  12641  12648  12652  12654  12665  12669
                   12671  12681  12685  12687  12740  12746  12750  12752  12762  12766  12767*  12768  12778
                   12782  12784  13835
RMDSO   001412      1411#
RMDT  = 000026      1123#   3579
RMDTI   001356      1390#   3581*  3586   3588   3590   3592  13835
RMDTO   001426      1417#
RMEC1 = 000044      1130#
RMEC1I  001374      1397#   9995  13835
RMEC1O  001444      1424#
```

```
RMEC2 = 000046          1131# 10078
RMEC2I  001376          1398# 10016  10077  11547  12111  13835
RMEC20  001446          1425#
RMER1 = 000014          1119#  7297   7389  10452
RMER1I  001344          1385#  6443   6453   6462   6473   6574   6584   6593   6604   6841   6843   6932   6934
                        9651   9984   9986  10444  10459  10461  10551  10627  10654  10697  10712  10755  10832
                       10858  10873  10888  10941  10960  10963  11051  11066  11081  11092  11220  11227  11254
                       11328  11331  11352  11524  11648  11652  11654  11655  11666  11668  11669  11680  11682
                       11683  11694  11696  11697  11708  11710  11711  11754  11759  11763  11765  11775  11777
                       11779  11791  11793  11795  11808  11810  11812  11885  11962  11966  11968  11970  11980
                       11982  11984  11994  11998  12000  12058  12180  12184  12186  12223  12225  12227  12268
                       12271  12273  12275  12285  12287  12289  12299  12301  12303  12326  12416  12419  12423
                       12425  12434  12436  12438  12448  12450  12451  12485  12489  12491  12493  12502  12504
                       12506  12515  12517  12519  12528  12530  12532  12550  12552  12554  12560  12616  12618
                       12620  12667  12748  12859  13835
RMER10  001414          1412#  7299*  7391* 10448
RMER2 = 000042          1129#
RMER2I  001372          1396#  6479   6489   6610   6620   6846   6848   6937   6939   7121   7131   7132   7226
                        7235   7237   9653   9658  10446  10553  10671  10727  10903  11222  11273  11276  11287
                       11315  11318  11367  11396  11569  11761  11822  11829  11831  11833  11846  11848  11850
                       11860  11862  11864  11901  11948  11996  12120  12182  12251  12254  12256  12340  12344
                       12369  12373  12375  12377  12387  12391  12393  12403  12405  12407  12421  12603  12605
                       12607  12650  12683  12764  12780  12794  12800  12802  12804  12814  12816  12818  12862
                       13835
RMER20  001442          1423#
RMLA  = 000020          1121#
RMLAI   001350          1387# 13835
RMLA0   001420          1414#
RMMR1 = 000024          1122#  7059   7068   7165   7174
RMMR1I  001354          1389# 11534  12086  13835
RMMR10  001424          1416#  7061*  7070*  7167*  7176*
RMMR2 = 000040          1128#
RMMR2I  001370          1395# 11556  12098  13835
RMMR20  001440          1422#
RMOF  = 000032          1125#  3651   3717   3795   3908   3974   4051   4156   4222   4302   4448   4514   4592
                        4705   4771   4848   4953   5019   5099   5245   5316   5393   5498   5594   5700   5814
                        5885   5962   6067   6138   6215   6331   6397   6528   6664   6740   6806   6896   7000
                        7086   7190   7314   7405   7516   7587   7668   7781   7847   7926   8045   8111   8188
                        8293   8359   8437   8550   8616   8694   8807   8873   8951   9495
RMOFI   001362          1392#  9988   9990  10848  10958  12481  12557  13835
RMOF0   001432          1419#  3630*  3887*  4135*  4427*  4684*  4932*  5224*  5477*  5793*  6046*  6303*  6650*
                        6712*  6800*  6987*  7031*  7270*  7495*  7760*  8024*  8272*  8529*  8786*  9488*  9917
                       11247  12323  13835
RMR   = 000004           995# 11648  11680  11684  11962  11980  11983  12268  12299  12302
RMSN  = 000030          1124#
RMSNI   001360          1391# 13835
RMSN0   001430          1418#
RMWC  = 000002          1201#  3495*  3496   3650   3719   3798   3907   3976   4054   4155   4224   4305   4447
                        4516   4595   4704   4773   4851   4952   5021   5102   5244   5318   5396   5497   5596
                        5703   5813   5887   5965   6066   6140   6218   6330   6399   6531   6667   6739   6808
                        6899   7003   7088   7193   7316   7408   7515   7589   7671   7780   7849   7929   8044
                        8113   8191   8292   8361   8440   8549   8618   8697   8806   8875   8954   9494
RMWCI   001332          1380# 10033  10982  10994  11012  11076  13835
RMWC0   001402          1407#  3629*  3712*  3886*  3969*  4134*  4216*  4426*  4509*  4683*  4766*  4931*  5013*
                        5223*  5285*  5476*  5588*  5792*  5854*  6045*  6108*  6302*  6392*  6524*  6649*  6711*
                        6996*  7030*  7269*  7494*  7580*  7759*  7841*  7922*  8023*  8105*  8271*  8353*  8528*
```

```
                    8610*  8785*  8868*  9487*  9749   9908  10034  10995  11013
RQA    = 100000     1080*  11557  12099
RQB    = 040000     1081*  11557  12099
RTC    = 000016      922*
SADMSK= 000037       964*
SAVREG= 104414      9094   13732*
SA1    = 000001      959*
SA16   = 000020      955*
SA2    = 000002      958*
SA4    = 000004      957*
SA8    = 000010      956*
SC     = 100000     1152*  10399  10416  10432  11485
SCTMSG  067466      3637    3894   4142   4434   4691   4939   5231   5484   5800   6053   6310   6719   7502
                    7767    8031   8279   8536   8793  13835*
SCTMSK= 003700       1015*
SC0    = 000100      1013*
SC1    = 000200      1012*
SC2    = 000400      1011*
SC3    = 001000      1010*
SC4    = 002000      1009*
SEARCH= 000030       928*  10575  10976
SECERR  045364      3761    3840   4018   4096   4266   4381   4558   4637   4815   4893   5063   5178   5360
                    5438    5635   5743   5929   6007   6182   6260   6494   6625   6863   6954   7137   7242
                    7362    7456   7631   7713   7891   7971   8155   8233   8403   8482   8660   8739   8917
                    8996   10495*
SEEK   = 000004      917*   5559   5667
SEKSTS  051440      5576    5684  11209*
SHUT    063404      9075   12928*
SHUT2   063464     12943*  13432  13515
SIZCLK  044224      3450   10242*
SKI    = 040000     1101*  11239  11242  11258  11288  11290  11294  11305  11367  11570  11822  11846  11849
                   11948   12313  12316  12330  12341  12343  12344  12346  12358  12650  12780  12795  12814
                   12817
SNGPRT= 020024       1063*  3586   3590   3594
STACK  = 001100       792*  3191   3484   3524   3568   3606   3863   4111   4403   4660   4908   5200   5453
                    5769    6022   6275   6686   7021   7259   7471   7736   8000   8248   8505   8762
STANDA  006120      3260    3295*
START   005324      1238    3175*  3513   12937
STCDRV  062436     12733*
STKLMT= 177774       803*
STOP    063354     12911*  13253
STSD1   105420     13835*
STSD2   105436     13835*
STSD3   105456     13835*
STSD4   105470     13835*
STSF    105521     13835*
STSH1   105063     13835*
STSH2   105140     13835*
STSH3   105227     13835*
STSH4   105265     13835*
SWR     001154      1291*  3189   3211*  3213   3219*  3226*  3240   3461   9095   13254  13268  13271  13286
                   13293   13355  13362  13374  13378  13435  13474  13529* 13746  13759*
SWREG   000176      1220*  3219   3240   3461   13435  13474  13497
SW0    = 000001      856*
SW00   = 000001      846*   856
SW01   = 000002      845*   855
```

```
SW02  = 000004            844#   854
SW03  = 000010            843#   853
SW04  = 000020            842#   852
SW05  = 000040            841#   851
SW06  = 000100            840#   850
SW07  = 000200            839#   849
SW08  = 000400            838#   848
SW09  = 001000            837#   847
SW1   = 000002            855#
SW10  = 002000            836#
SW11  = 004000            835#
SW12  = 010000            834#
SW13  = 020000            833#  9095
SW14  = 040000            832#
SW15  = 100000            831#
SW2   = 000004            854#
SW3   = 000010            853#
SW4   = 000020            852#
SW5   = 000040            851#
SW6   = 000100            850#
SW7   = 000200            849#
SW8   = 000400            848#
SW9   = 001000            847#
TADMSK= 003400            963#
TAG   = 020000           1082#
TAGADR= 001114           1263#  1270
TAP   = 040000           1059#
TA1   = 000400            954#
TA2   = 001000            953#
TA4   = 002000            952#
TBITVE= 000014            889#
TIMOUT  044346           3668   3734   3813   3925   3991   4069   4173   4239   4320   4465   4531   4610   4722
                         4788   4866   4970   5036   5117   5262   5333   5411   5515   5569   5608   5677   5716
                         5831   5902   5980   6084   6155   6233   6348   6414   6546   6676   6757   6823   6914
                         7012   7103   7208   7343   7438   7533   7604   7686   7798   7864   7944   8062   8128
                         8206   8310   8376   8455   8567   8633   8712   8824   8890   8969   9405   9583   9617
                        10271#
TKVEC = 000060            896#  13412*  13413*
TPVEC = 000064            897#
TRAPVE= 000034            895#   3197*  3198*
TRE   = 040000           1153#  10399   10416   10432   10577   10579   10984
TRTVEC= 000014            890#
TST   = 010000           1083#  11558   12100
TSTNMB  037752           9106*   9107*   9109   9226#
TSTPRP  040020           3617   3699   3780   3874   3956   4037   4122   4204   4288   4414   4496   4577   4671
                         4753   4834   4919   5001   5085   5211   5300   5379   5464   5549   5657   5780   5869
                         5948   6033   6122   6201   6290   6379   6512   6654   6699   6788   6881   6990   7050
                         7156   7288   7381   7482   7564   7650   7747   7829   7910   8011   8093   8174   8259
                         8341   8422   8516   8598   8679   8773   8855   8936   9274#
TSTQUE  001450           1431#  3433   3434*  3474   3486   3526   3570   3608   3865   4113   4405   4662   4910
                         5202   5455   5771   6024   6277   6688   7023   7261   7473   7738   8002   8250   8507
                         8764   9022   9024*  9028*  11157  11442  11509  12070
TST1    007136           3479#  13313
TST10   014200           4657#  13320
TST11   015172           4905#  13321
TST12   016434           5197#  13322
```

```
TST13   017454          5450#  13323
TST14   021002          5766#  13324
TST15   022022          6019#  13325
TST16   023042          6272#  13326
TST17   024740          6683#  13327
TST2    007334          3521#  13314
TST20   026414          7018#  13328
TST21   027522          7256#  13329
TST22   030470          7468#  13330
TST23   031546          7733#  13331
TST24   032634          7997#  13332
TST25   033626          8245#  13333
TST26   034650          8502#  13334
TST27   035672          8759#  13335
TST3    007520          3565#  13315
TST4    007702          3603#  13316
TST5    010724          3860#  13317
TST6    011716          4108#  13318
TST7    013156          4400#  13319
TYPBN = 104406          9216   13724#
TYPDS = 104405          9056    9063   9210  13723#
TYPE  = 104401          3234    3254   3282   3304   3307   3313   3318   3321   3324   3328   3331   3339   3342
                        3348    3353   3358   3364   3374   3379   3385   3398   3399   3402   3408   3411   3418
                        3637    3894   4142   4434   4691   4939   5231   5484   5800   6053   6310   6719   7502
                        7767    8031   8279   8536   8793   9050   9057   9064   9099   9100   9108   9117   9123
                        9132    9170   9176   9180   9193   9199   9201   9219   9221  12932  13006  13072  13141
                       13201   13357  13365  13429  13441  13495  13496  13499  13510  13520  13531  13550  13598
                       13604   13609  13613  13618  13619  13621  13624  13628  13719#  13768
TYPOC = 104402          3341    9213   13498  13720#
TYPON = 104404         13722#
TYPOS = 104403          3284    3355   3376   9103   9111   9120   9126  13721#
UBUSQS  067775          3328   13835#
UNS   = 040000           983#   7299   7391  11525  11648  11652  11656  11962  11994  11999  12416  12419  12424
UNTMSK= 000007          1184#  10357  10359
UPE   = 020000          1166#  10741
USE   = 040000          1112#   6438   6569   9692   9915
USRFIL  106540          9706*   9713*  9864   9866  13835#
U0    = 000001          1181#   7577   7662
U1    = 000002          1180#   7577   7662
U2    = 000004          1179#   7577   7662
VV    = 000100           977#   9342   9537  10667  11279  11344  11345  11394  11399  11620  11623  11836  11875
                       11876  11899  11905  12046  12230  12259  12642  12643  12681  12686  12741  12742  12762
                       12767
WC    = 000040          1049#  11535  12087
WCD   = 000050           937#   4048   4299   4845   5096   5390   5959   6212   8185
WCE   = 040000          1165#   4338   4348   5135   5145  10640  13835
WCEHI = 010000          1190#
WCELO = 004000          1191#
WCF   = 000040           992#    999  10711  12616  12619
WCH   = 000052           938#
WD    = 000060           941#   3713   3970   4218   4510   4767   5015   5287   5312   5590   5858   5881   6107
                       6134   6393   6801   7032   7081   7310   7582   7843   8107   8355   8612   8869
WH    = 000062           942#   3631   3888   4136   4428   4685   4933   5225   5478   5794   6047   6304   6651
                       6661   6713   6988   6997   7271   7496   7761   8025   8273   8530   8787  12479
WLE   = 004000           986#    999  10692  10696  10709  10724  10739  12416  12448  12452  12460  13835
WRL   = 004000           972#  10694  12455
```

```
XDPMG   070565            3254  13835#
XSIZ    005776            3264#  3405
ZEROS   071110            3641   3898   4146   5235   5856   6057   6314   7771  13835#
$APTHD  001100            1250   1256#
$ASTAT= ****** U         13811  13826
$ATYC   067244           13782  13784#
$ATY1   067220           13780#
$ATY3   067226           13186  13781#
$ATY4   067236           13370  13783#
$AUTOB  001150            1288#  3244#  3465#  13488  13645
$BASE   001276            1357#  3270   3340   3350#  3485   3525   3569   3607   3864   4112   4404   4661   4909
                          5201   5454   5770   6023   6276   6687   7022   7260   7472   7737   8001   8249   8506
                          8763  10129  10202  10279  11156  11440  13835
$BDADR  001136            1283#  4369#  4370#  4371   5166#  5167#  5168  10044#  13835
$BDDAT  001142            1285#  4349#  4365#  4375   5146#  5162#  5172   6453#  6473#  6489#  6584#  6604#  6620#
                          6843#  6844#  6848#  6849#  6934#  6935#  6939#  6940#  7131#  7237#  7338#  7432#  10042#
                         10356# 10357# 10360  10378# 10384# 10400# 10417# 10433# 10461# 10511# 10512# 10524# 10525#
                         10535# 10536# 10538  10556# 10557# 10558  10578# 10579# 10580  10611# 10612# 10613  10627#
                         10628# 10629  10641# 10642# 10654# 10655# 10671# 10672# 10697# 10698# 10712# 10713# 10727#
                         10728# 10742# 10743# 10755# 10756# 10785# 10786# 10800# 10801# 10815# 10816# 10832# 10833#
                         10858# 10859# 10873# 10874# 10888# 10889# 10903# 10904# 10926# 10927# 10941# 10942# 10963#
                         10964# 10982# 11003# 11054# 11055# 11056  11081# 11082# 11083  11097# 11098  11109# 11110
                         11227# 11254# 11255# 11256  11276# 11287# 11288# 11294  11305  11318# 11331# 11356# 11370#
                         11371# 11384# 11400# 11484# 11485# 11487  11497# 11506# 11514  11524# 11525# 11534# 11535#
                         11537  11547# 11556# 11557# 11559  11569# 11570# 11579# 11580# 11582  11624# 11638# 11654#
                         11668# 11682# 11696# 11710# 11765# 11779# 11795# 11812# 11833# 11850# 11864# 11889# 11904#
                         11919# 11935# 11952# 11970# 11984# 12000# 12033# 12034# 12036  12045# 12046# 12048  12058#
                         12067# 12074# 12077  12086# 12087# 12089  12098# 12099# 12101  12111# 12120# 12170# 12186#
                         12207# 12227# 12256# 12275# 12289# 12303# 12326# 12327# 12328  12340# 12341# 12377# 12393#
                         12407# 12425# 12438# 12450# 12493# 12506# 12519# 12532# 12554# 12579# 12594# 12607# 12620#
                         12633# 12654# 12671# 12687# 12752# 12768# 12784# 12804# 12818# 12874# 12889# 13835
$BELL   001212            1308#  13357  13390  13441  13638
$BIN    063642           13002# 13005# 13006  13013#
$CDW1   001302            1359#
$CDW2   001304            1360#
$CHARC  064574           13203# 13213# 13220  13229# 13234#
$CKSWR  065672           13474# 13728
$CMTAG  001114            1271#  3186   3187   3195   3201   3202   3203
$CM3    =  000000         1301#
$CM4    =  000005         1301#  1302#  1303#  1304#  1305#  1306#
$CNTLC  066570           13429  13510  13638#
$CNTLG  066602           13495  13640#
$CNTLU  066575           13520  13613  13639#
$CPUOP  001250            1331#
$CRLF   001217            1310#  9064   9099   9132   9170   9180   9193   9201   9221  13202  13237  13365  13390
                         13531  13618  13638
$DBLK   064060           13038  13072  13080#
$DDW0   001306            1361#
$DDW1   001310            1362#
$DDW2   001312            1363#
$DDW3   001314            1364#
$DDW4   001316            1365#
$DDW5   001320            1366#
$DDW6   001322            1367#
$DDW7   001324            1368#
$DEVCT  001232            1322#
```

```
$DEVM   001300  1358#  3267*  3279*  3397*  3406*  3424*  3425  3432
$DOAGN  037206  9046  9067  9073#
$DTBL   064050  13041  13076#
$ENDAD  037176  1229  3232  9069#  13385
$ENDCT  037042  3201  9048#
$ENULL  037212  9076#
$ENV    001242  1327#  3238  3259  3459  13181  13367  13789  13813
$ENVM   001243  1328#  3224  3264  13183  13188  13791
$EOP    037006  3514  9038#  12935
$EOPCT  037034  3201*  9045#  9049
$EOSP   036752  3557  3597  9020#
$ERFLG  001117  1274#  13242  13282  13284  13290*  13312  13352*  13390
$ERMAX  001131  1280#  3204*  13284  13307*  13312
$ERROR  065174  3195  13350#
$ERRPC  001132  1281#  9124  13359*  13360*  13361  13390
$ERRTB  001564  1470#  9139
$ERTTL  001126  1278#  9061  9065*  13358*  13390
$ESCAP  001210  1307#  3203*  4277*  4394*  5074*  5191*  5538*  5583*  5647*  5691*  13306*  13381  13383
                13390
$ETABL  001242  1326#
$ETEND  001326  1262  1369#
$FATAL  001224  1319#  13817*
$FFLG   067464  13780*  13783*  13811  13820*  13828#
$FILLC  001172  1299#  13206  13237
$FILLS  001171  1298#  13237
$GDADR  001134  1282#  4368*  4371  5165*  5168  10043*  13835
$GDDAT  001140  1284#  4347*  4348*  4366*  4367*  4375  5144*  5145*  5163*  5164*  5172  6454*  6472*
                6490*  6585*  6603*  6621*  6850*  6941*  7132*  7133*  7235*  7236*  7336*  7337*  7430*
                7431*  10041*  10358*  10359*  10360  10376*  10377*  10383*  10385*  10397*  10398*  10399*  10415*
                10416*  10431*  10432*  10459*  10460*  10513*  10526*  10537*  10538  10550*  10555*  10558  10569*
                10577*  10580  10606*  10609*  10610*  10613  10625*  10626*  10629  10636*  10988*  10994*  10995*
                10996*  10997*  11000*  11001  11048*  11053*  11056  11065*  11078*  11083  11094*  11095*  11096*
                11098  11107*  11108*  11110  11226*  11238*  11251*  11256  11275*  11289*  11290*  11317*  11330*
                11354*  11355*  11369*  11382*  11383*  11398*  11399*  11486*  11487  11496*  11512*  11514  11523*
                11536*  11537  11546*  11558*  11559  11568*  11581*  11582  11622*  11623*  11636*  11637*  11655*
                11656*  11669*  11670*  11683*  11684*  11697*  11698*  11711*  11712*  11763*  11764*  11777*  11778*
                11793*  11794*  11810*  11811*  11831*  11832*  11848*  11849*  11862*  11863*  11887*  11888*  11903*
                11905*  11917*  11918*  11933*  11934*  11950*  11951*  11968*  11969*  11982*  11983*  11998*  11999*
                12035*  12036  12047*  12048  12057*  12088*  12089  12100*  12101  12109*  12168*  12169*  12184*
                12185*  12205*  12206*  12225*  12226*  12254*  12255*  12273*  12274*  12287*  12288*  12301*  12302*
                12312*  12325*  12328  12342*  12343*  12375*  12376*  12391*  12392*  12405*  12406*  12423*  12424*
                12436*  12437*  12451*  12452*  12460*  12491*  12492*  12504*  12505*  12517*  12518*  12530*  12531*
                12552*  12553*  12577*  12578*  12592*  12593*  12605*  12606*  12618*  12619*  12631*  12632*  12652*
                12653*  12669*  12670*  12685*  12686*  12750*  12751*  12766*  12782*  12783*  12802*  12803*  12816*
                12817*  12875*  12890*  13835
$GET42  037166  9066#
$GTSWR  065762  13496#  13726
$HD   = 000001  768  769
$HIBTS  001100  1257#
$HIOCT  066732  13678*  13683#
$ICNT   001120  1275#  13297*  13298  13300*  13311
$ILLUP  067202  13738  13754  13771#
$INTAG  001151  1289#  13493*  13512  13532  13645
$ITEMB  001130  1279#  9115  13361*  13369  13390
$LF     001220  1311#  13237  13390  13628  13638
$LFLG   067463  13821*  13827#
```

```
$LPADR  001122          1276# 3205# 3481# 3688# 3770# 3945# 4027# 4193# 4275# 4395# 4485# 4567# 4742#
                        4824# 4990# 5072# 5192# 5282# 5369# 5535# 5644# 5851# 5938# 6104# 6191# 6368#
                        6503# 6678# 6777# 6872# 7013# 7039# 7146# 7251# 7277# 7371# 7553# 7640# 7818#
                        7900# 7992# 8082# 8164# 8330# 8412# 8587# 8669# 8844# 8926# 9017# 13279# 13288#
                       13304# 13309 13311
$LPERR  001124          1277# 3206# 3482# 3689# 3771# 3946# 4028# 4194# 4276# 4396# 4486# 4568# 4743#
                        4825# 4991# 5073# 5193# 5283# 5370# 5536# 5645# 5852# 5939# 6105# 6192# 6369#
                        6504# 6679# 6778# 6873# 7014# 7040# 7147# 7252# 7278# 7372# 7554# 7641# 7819#
                        7901# 7993# 8083# 8165# 8331# 8413# 8588# 8670# 8845# 8927# 9018# 13288  13305#
                       13311  13380
$MADR1  001254          1344#
$MADR2  001260          1348#
$MADR3  001264          1351#
$MADR4  001270          1354#
$MAIL   001222          1258   1262   1317#  3223   3238   3459   3487   3527   3571   3609   3866   4114   4406
                        4663   4911   5203   5456   5772   6025   6278   6689   7024   7262   7474   7739   8003
                        8251   8508   8765  13181  13303  13367
$MAMS1  001252          1338#
$MAMS2  001256          1346#
$MAMS3  001262          1349#
$MAMS4  001266          1352#
$MBADR  001102          1258#
$MFLG   067462         13781* 13787  13822* 13826#
$MNEW   066620         13499  13643#
$MSGAD  001236          1324# 13797* 13800
$MSGLG  001240          1325# 13802#
$MSGTY  001222          1318# 13795  13803* 13815  13819*
$MSWR   066607         13496  13641#
$MTYP1  001253          1339#
$MTYP2  001257          1347#
$MTYP3  001263          1350#
$MTYP4  001267          1353#
$MXCNT  065114         13301  13311#
$NULL   001170          1297# 13208  13237
$NWTST= 000001          3476#  3478   3517#  3519   3561#  3563   3600#  3857#  4105#  4397#  4654#  4902#  5194#
                        5447#  5763#  6016#  6269#  6680#  7015#  7253#  7465#  7730#  7994#  8242#  8499#  8756#
$OCNT   064312         13113# 13142# 13155#
$OMODE  064314         13108# 13112# 13117  13120# 13131# 13157#
$OVER   065100         13255  13280  13289  13299  13308#
$PASS   001230          1321#  3223#  9042#  9043#  9054   9076  13295  13312
$PASTM  001106          1260#
$POWER  067210         13769  13774#
$PWRDN  067042          3199  13738#  13766
$PWRMG  067176         13769#
$PWRUP  067114         13748  13754#
$QUES   001216          1309# 13237  13390  13550  13621  13638
$RDCHR  066234         13563# 13729
$RDDEC= ****** U       13732
$RDLIN  066324         13586# 13730
$RDOCT  066632         13656# 13731
$RDSZ = 000010         13579#
$RESRE  063532         12978# 13733
$RTNAD  037210          9075#
$R2A  = ****** U       13734
$SAVRE  063474         12962# 13732
$SAVR6  067206         13747# 13755  13756# 13757# 13773#
```

```
$SCOPE  064600          3193  13251#
$SETUP= 000137          1211#  3192   3193   3195   3197   3199   3201   3202   3203   3205   3232   3235   3456
                        9040  13252  13351  13377  13385  13433  13438  13439  13469  13645
$STUP = 177777          1211#
$SVLAD  065044         13263  13302#
$SVPC = 000200          1227#  1232
$SWR  = 167400           757#   768    773    774    775    776    777    778    779    780   1306   1307   1308
                        3202   3203   3205   3206   3487   3527   3571   3609   3866   4114   4406   4663   4911
                        5203   5456   5772   6025   6278   6689   7024   7262   7474   7739   8003   8251   8508
                        8765   9035   9041   9068   9074   9076  13243  13244  13245  13246  13247  13254  13266
                       13268  13269  13282  13283  13284  13291  13292  13293  13305  13308  13311  13342  13343
                       13344  13345  13346  13355  13362  13374  13378  13390  13770
$SWREG  001244          1329#  3226
$SWRMK= 000000           780   781  13247  13248  13272
$SWO8T  065116         13278  13312#
$TESTN  001226          1320#  3487*  3527*  3571*  3609*  3866*  4114*  4406*  4663*  4911*  5203*  5456*  5772*
                        6025*  6278*  6689*  7024*  7262*  7474*  7739*  8003*  8251*  8508*  8765*  9107  13303*
$TIMES  001206          1306#  3202*  9041*  13291*  13298  13301*  13311
$TKB    001162          1294#  13393  13414  13425  13450  13478  13505
$TKCNT  065374         13394#  13409*  13439  13456*  13570  13572*
$TKINT  065404          3296   3469  13409#  13430  13491
$TKQEN= 065403         13398#  13464  13575
$TKQIN  065376         13395#  13410*  13411  13462*  13463*  13464  13466*
$TKQOU  065400         13396#  13411*  13573  13574*  13575  13577*
$TKQSR  065402         13397#  13410  13466  13577
$TKS    001160          1293#  13393  13415*  13446*  13448  13454*  13476  13492*  13502  13514*  13534*
$TKSRV  065454         13412  13425#
$TMP0   001174          1301#   3541*  3544   3641*  3898*  4146*  4350*  4438*  4695*  4943*  5147*  5235*  5288*
                        5488*  5804*  5856*  6057*  6110*  6314*  6455*  6586*  6723*  7506*  7771*  8035*  8283*
                        8540*  8797*  9938  10045*  10134*  10207*  11162*  11163  13835
$TMP1   001176          1302#  3306*  3307   3308   3311   3320*  3321   3322   3330*  3331   3334   3344*  3346
                        3350   3351*  3352*  3354   3360*  3362   3366   3367*  3368*  3369*  3370*  3371*  3372*
                        3373*  3375   3381*  3383   3388*  3389*  3390*  3391*  3392*  3393   3401*  3402   3403
                        3410*  3411   3412   3414   3416   3420   3540*  3548   3594*  3642*  3899*  4147*  4351*
                        4439*  4696*  4944*  5148*  5236*  5289*  5489*  5805*  5857*  6058*  6111*  6315*  6456*
                        6587*  6724*  7507*  7772*  8036*  8284*  8541*  8798*  9939  10135*  10136  10208*  10209
                       11161*  11168  12859*  12860*  12872  12874  13835
$TMP2   001200          1303#   3595*  12862*  12863*  12886  12889  13835
$TMP3   001202          1304#
$TMP4   001204          1305#
$TN   = 000030           768#   3476   3487#  3517   3527#  3561   3571#  3600   3609#  3857   3866#  4105   4114#
                        4397   4406#  4654   4663#  4902   4911#  5194   5203#  5447   5456#  5763   5772#  6016
                        6025#  6269   6278#  6680   6689#  7015   7024#  7253   7262#  7465   7474#  7730   7739#
                        7994   8003#  8242   8251#  8499   8508#  8756   8765#  13273  13313
$TPB    001166          1296#  13226*  13237
$TPFLG  001173          1300#  13175   13237
$TPS    001164          1295#  13224   13237
$TRAP   066734          3197   13692#
$TRAP2  066774         13707#  13718
$TRP  = 000016         13711#  13720#  13721#  13722#  13723#  13724#  13725#  13726  13727#  13728  13729#  13730#  13731#
                       13732#  13733#  13734#
$TRPAD  067006         13701   13718#
$TSTM   001104          1259#
$TSTNM  001116          1273#   9040*  13242  13275*  13302*  13303  13308  13312  13354  13390
$TTYIN  066560         13588   13589  13601  13619  13633  13637#
$TYPBN  063570         12999#  13724
```

```
$TYPDS  063644          13026#  13723
$TYPE   064316          13175#  13711   13719   13808
$TYPEC  064530          13205   13212   13219   13224#  13225   13536
$TYPEX  064576          13230   13232   13235#
$TYPOC  064114          13111#  13720
$TYPON  064130          13110   13113#  13722
$TYPOS  064070          13106#  13721
$UNIT   001234           1323#   3474*   9101   10358
$UNITM  001110           1261#
$USWR   001246           1330#
$VECT1  001272           1355#   3351    3366*   3367    3393*
$VECT2  001274           1356#
$XTSTR  064616          13257#
$$GET4= 000000           9068#
$$SW08= 000030          13312#  13313   13314#  13315#  13316#  13317#  13318#  13319#  13320#  13321#  13322#  13323#  13324#
                        13325#  13326#  13327#  13328#  13329#  13330#  13331#  13332#  13333#  13334#  13335#  13336#
$OFILL  064313          13107*  13111*  13121   13156#
$40CAT= ****** U        13254   13364
.       = 111753         1214#   1218#   1227    1228#   1230#   1232#   1237#   1240#   1246    1247#   1249#   1251#   1263
                         1270#   1312    1431#   1447#   1452#   3190    3205    3206    3248#   9053#   9076    9077#  12930
                        12942#  13080#  13237   13311   13312   13390   13393   13397#  13398   13399#  13637#  13638   13645#
                        13750   13772   13829#  13835#
.$ASTA= ****** U        13781   13784
.$X     = 001100         1246#   1251
```

CALCLR 757# 3529 3573
CALGET 757#
CALPUT 757# 6670 7006
CALSUB 757# 3617 3635 3657 3665 3668 3671 3679 3699 3723 3731 3734 3737 3745 3752
3761 3780 3802 3810 3813 3816 3824 3831 3840 3847 3874 3892 3914 3922 3925
3928 3936 3956 3980 3988 3991 3994 4002 4009 4018 4037 4058 4066 4069 4072
4080 4087 4096 4122 4140 4162 4170 4173 4176 4184 4204 4228 4236 4239 4242
4250 4257 4266 4288 4309 4317 4320 4323 4331 4340 4381 4414 4432 4454 4462
4465 4468 4476 4496 4520 4528 4531 4534 4542 4549 4558 4577 4599 4607 4610
4613 4621 4628 4637 4644 4671 4689 4711 4719 4722 4725 4733 4753 4777 4785
4788 4791 4799 4806 4815 4834 4855 4863 4866 4869 4877 4884 4893 4919 4937
4959 4967 4970 4973 4981 5001 5025 5033 5036 5039 5047 5054 5063 5085 5106
5114 5117 5120 5128 5137 5178 5211 5229 5251 5259 5262 5265 5273 5300 5322
5330 5333 5336 5344 5351 5360 5379 5400 5408 5411 5414 5422 5429 5438 5464
5482 5504 5512 5515 5518 5526 5549 5560 5568 5569 5570 5576 5600 5608 5611
5619 5626 5635 5657 5668 5676 5677 5678 5684 5707 5716 5719 5727 5734 5743
5750 5780 5798 5820 5828 5831 5834 5842 5869 5891 5899 5902 5905 5913 5920
5929 5948 5969 5977 5980 5983 5991 5998 6007 6033 6051 6073 6081 6084 6087
6095 6122 6144 6152 6155 6158 6166 6173 6182 6201 6222 6230 6233 6236 6244
6251 6260 6290 6308 6337 6345 6348 6351 6359 6379 6403 6411 6414 6417 6425
6446 6465 6482 6494 6512 6535 6543 6546 6549 6557 6577 6596 6613 6625 6654
6699 6717 6746 6754 6757 6760 6768 6788 6812 6820 6823 6826 6834 6854 6863
6881 6903 6911 6914 6917 6925 6945 6954 6961 6990 7035 7050 7092 7100 7103
7106 7114 7123 7137 7156 7197 7205 7208 7211 7219 7228 7242 7274 7288 7322
7323 7329 7343 7346 7354 7362 7381 7413 7415 7423 7438 7441 7449 7456 7482
7500 7522 7530 7533 7536 7544 7564 7593 7601 7604 7607 7615 7622 7631 7650
7675 7683 7686 7689 7697 7704 7713 7720 7747 7765 7787 7795 7798 7801 7809
7829 7853 7861 7864 7867 7875 7882 7891 7910 7933 7941 7944 7947 7955 7962
7971 7978 8011 8029 8051 8059 8062 8065 8073 8093 8117 8125 8128 8131 8139
8146 8155 8174 8195 8203 8206 8209 8217 8224 8233 8259 8277 8299 8307 8310
8313 8321 8341 8365 8373 8376 8379 8387 8394 8403 8422 8444 8452 8455 8458
8466 8473 8482 8489 8516 8534 8556 8564 8567 8570 8578 8598 8622 8630 8633
8636 8644 8651 8660 8679 8701 8709 8712 8715 8723 8730 8739 8746 8773 8791
8813 8821 8824 8827 8835 8855 8879 8887 8890 8893 8901 8908 8917 8936 8958
8966 8969 8972 8980 8987 8996 9003
CNACMS 3476# 3478
COMMEN 899#
DVAVMS 3517# 3519
DVTYMS 3561# 3563
ENDCOM 899#
ERROR 793# 3452 3510 3532 3546 3550 3556 3576 3585 3596 3621 3638 3660 3674 3682
3703 3726 3740 3748 3755 3764 3784 3805 3819 3827 3834 3843 3852 3878 3895
3917 3931 3939 3960 3983 3997 4005 4012 4021 4041 4061 4075 4083 4090 4099
4126 4143 4165 4179 4187 4208 4231 4245 4253 4260 4269 4292 4312 4326 4334
4343 4352 4361 4373 4377 4384 4418 4435 4457 4471 4479 4500 4523 4537 4545
4552 4561 4581 4602 4616 4624 4631 4640 4649 4675 4692 4714 4728 4736 4757
4780 4794 4802 4809 4818 4838 4858 4872 4880 4887 4896 4923 4940 4962 4976
4984 5005 5028 5042 5050 5057 5066 5089 5109 5123 5131 5140 5149 5159 5170
5174 5181 5215 5232 5254 5268 5276 5304 5325 5339 5347 5354 5363 5383 5403
5417 5425 5432 5441 5468 5485 5507 5521 5529 5553 5563 5573 5579 5603 5614
5622 5629 5638 5661 5671 5681 5687 5710 5722 5730 5737 5746 5755 5784 5801
5823 5837 5845 5873 5894 5908 5916 5923 5932 5952 5972 5986 5994 6001 6010
6037 6054 6076 6090 6098 6126 6147 6161 6169 6176 6185 6205 6225 6239 6247
6254 6263 6294 6311 6340 6354 6362 6383 6406 6420 6428 6449 6457 6468 6474
6485 6491 6497 6516 6538 6552 6560 6580 6588 6599 6605 6616 6622 6628 6658
6673 6703 6720 6749 6763 6771 6792 6815 6829 6837 6851 6857 6866 6885 6906

```
            6920    6928    6942    6948    6957    6966    6994    7009    7054    7065    7074    7095    7109    7117    7126
            7134    7140    7160    7171    7180    7200    7214    7222    7231    7238    7245    7292    7303    7326    7332
            7339    7349    7357    7365    7385    7395    7418    7426    7433    7444    7452    7459    7486    7503    7525
            7539    7547    7568    7596    7610    7618    7625    7634    7654    7678    7692    7700    7707    7716    7725
            7751    7768    7790    7804    7812    7833    7856    7870    7878    7885    7894    7914    7936    7950    7958
            7965    7974    7983    8015    8032    8054    8068    8076    8097    8120    8134    8142    8149    8158    8178
            8198    8212    8220    8227    8236    8263    8280    8302    8316    8324    8345    8368    8382    8390    8397
            8406    8426    8447    8461    8469    8476    8485    8494    8520    8537    8559    8573    8581    8602    8625
            8639    8647    8654    8663    8683    8704    8718    8726    8733    8742    8751    8777    8794    8816    8830
            8838    8859    8882    8896    8904    8911    8920    8940    8961    8975    8983    8990    8999    9008
ESCAPE       899#    4277    4393    5074    5190    5538    5583    5647    5691
GETCS1       757#
GETDB        757#    4356    5153
GETDS        757#
GETDT        757#    3579
GETMR1       757#
GETPRI       899#
GETSWR       757#     899#    3235#   3456
MULT         899#
NEWTST       899#    3476    3517    3561    3600    3857    4105    4397    4654    4902    5194    5447    5763    6016    6269
            6680    7015    7253    7465    7730    7994    8242    8499    8756
NWTST        757#    3479    3521    3565    3603    3860    4108    4400    4657    4905    5197    5450    5766    6019    6272
            6683    7018    7256    7468    7733    7997    8245    8502    8759
POP          899#    3503    3504    3508    3509    3542    3543    3554    3555    9499    9718    9719    9732    9875    9876
            9877    9949    9951    9952    9953    9954   10055   10057   10058   10059   10088   10089   10090   10163   10165
           10166   10167   10168   10169   10170   10230   10232   10233   10234   10235   10236   10237   10259   10261   10301
           10303   10304   10305   10306   10454   10458   10602   11186   11187   11188   11189   11449   11451   11452   11453
           11513   12075   12076   12983   13067   13679   13759   13760   13823   13824
PUSH         899#    3489    3490    3534    3536    9476    9701    9702    9725    9743    9744    9745    9901    9903    9904
            9905    9906    9977    9978    9979    9980   10072   10074   10075   10122   10123   10124   10125   10126   10127
           10128   10195   10196   10197   10198   10199   10200   10201   10243   10244   10272   10273   10274   10275   10276
           10450   10599   11150   11151   11152   11153   11433   11435   11436   11437   11507   12068   12069   12963   13026
           13658   13740   13746   13784   13786   13807
PUTCS1       757#
PUTDC        757#
PUTER1       757#    7297    7389
PUTMR1       757#    7059    7068    7165    7174
REPORT       899#
RGBFMC       757#    1377    1404
SCOPE        794#    3522    3566    3604    3861    4109    4401    4658    4906    5198    5451    5767    6020    6273    6684
            7019    7257    7469    7734    7998    8246    8503    8760
SETPRI       899#    3180    3470   12914   12920   13566
SETTRA     13711#   13720   13721   13722   13723   13724   13726   13728   13729   13730   13731   13732   13733
SETUP        899#    3185
SKIP         899#
SLASH        899#
SPACE        899#
STARS        899#    1225    1243    1245    1252    1266    1312    1315    3476    3478    3517    3520    3561    3564    3600
            3602    3611    3692    3774    3857    3859    3868    3949    4031    4105    4107    4116    4197    4279    4397
            4399    4408    4489    4571    4654    4656    4665    4746    4828    4902    4904    4913    4994    5076    5194
            5196    5205    5293    5373    5447    5449    5458    5542    5651    5763    5765    5774    5862    5942    6016
            6018    6027    6115    6195    6269    6271    6280    6284    6372    6506    6638    6680    6682    6691    6781
            6875    7015    7017    7043    7150    7253    7255    7281    7375    7465    7467    7476    7557    7644    7730
            7732    7741    7822    7904    7994    7996    8005    8086    8168    8242    8244    8253    8334    8416    8499
            8501    8510    8591    8673    8756    8758    8767    8848    8930    9031    9080    9281    9294    9306    9328
            9357    9379    9407    9471    9737   10497   10505   10590   10973   12947   12992   13016   13083   13160   13239
```

```
            13338   13392   13469   13484   13555   13579   13648   13686   13736   13752   13779
SWRSU         899#   3207#
TAGS          757#   1371
TRMTRP      13711#
TYPBIN        899#
TYPDEC        899#   9054    9061
TYPNAM        757#    899#   3228
TYPNUM        899#
TYPOCS        899#   3354    3375    9101    9109    9118    9124
TYPOCT        899#   3340   13497
TYPTXT        899#   9050    9057
XPER          757#   1474    1481    1488    1495    1502    1509    1516    1523    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    1769    1776
             1783    1790    1797    1805    1813    1821    1828    1835    1842    1849    1856    1863    1870    1877    1885
             1892    1899    1906    1913    1920    1927    1934    1941    1948    1955    1962    1969    1976    1983    1990
             1997    2004    2011    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    2278    2285    2292    2299    2306
             2313    2320    2327    2334    2341    2348    2355    2362    2369    2376    2383    2390    2397    2404    2411
             2418    2425    2432    2439    2446    2453    2460    2467    2474    2481    2488    2495    2502    2509    2516
             2523    2530    2537    2544    2551    2558    2565    2572    2579    2586    2593    2600    2607    2614    2621
             2628    2635    2642    2649    2656    2663    2670    2677    2684    2691    2698    2705    2712    2719    2726
             2733    2740    2747    2755    2762    2770    2777    2784    2792    2800    2808    2817    2825    2833    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    3043    3050
             3057    3064    3071    3078    3085    3092    3099    3106    3113    3120    3127    3134
$$CMRE      1264#
$$CMTM      1264#   1301    1302    1303    1304    1305
$$ESCA        899#
$$NEWT        899#   3476    3517    3561    3600    3857    4105    4397    4654    4902    5194    5447    5763    6016    6269
             6680    7015    7253    7465    7730    7994    8242    8499    8756
$$SET       13711#  13720   13721   13722   13723   13724   13726   13728   13729   13730   13731   13732   13733
$$SETM       3223#
$$SKIP        899#
.EQUAT        757#     789
.HEADE        757#     758
.SETUP        757#    1211
.SWRHI        757#     769
.SWRLO        757#     781#
.$ACTI        757#    1223
.$APTB        757#    1313#
.$APTH        757#    1241
.$APTY        757#   13777
.$CATC        757#    1212
.$CMTA        757#    1264
.$DIV         757#
.$EOP         757#    9029
.$ERRO        757#   13336
.$ERRT        757#
.$MULT        757#
.$POWE        757#   13734
.$RAND        757#
.$RDDE        757#
.$RDOC        757#   13646
.$READ        757#   13390
```

```
.$SAVE    757#  12945
.$SCOP    757#  13237
.$SIZE    757#
.$TRAP    757#  13684
.$TYPB    757#  12990
.$TYPD    757#  13014
.$TYPE    757#  13158
.$TYPO    757#  13081

. ABS.  111753     000

 ERRORS DETECTED:  0

 RM03:CZRMEB.BIN,RM03:CZRMEB.SEQ/DOC/NL:TOC/SOL/CRF=RM03:CZRMEB.P11
 RUN-TIME:  100 97 4 SECONDS
 RUN-TIME RATIO: 918/201=4.5
 CORE USED:  34K  (67 PAGES)

 DOCUMENT PAGES:  332
```